Technical Instructions

IDC DOCUMENTATION

Global Association (GA) Subsystem Software User Manual

Approved for public release;
distribution unlimited

Notice

This document was published July 2001 by the Monitoring Systems Operation of Science Applications International Corporation (SAIC) as part of the International Data Centre (IDC) Documentation. Every effort was made to ensure that the information in this document was accurate at the time of publication. However, information is subject to change.

Contributors

Ronan J. LeBras, Science Applications International Corporation

Trademarks

Ordering Information

The ordering number for this document is SAIC-01/3003.

This document is cited within other IDC documents as [IDC6.5.12].

Global Association (GA) Subsystem Software User Manual

CONTENTS

Global Association (GA) Subsystem Software User Manual

FIGURES

Global Association (GA) Subsystem Software User Manual

TABLES

About this Document

This chapter describes the organization and content of the document and includes the following topics:

About this Document

PURPOSE

This document describes how to use the Global Association (GA) Subsystem software of the International Data Centre (IDC). The software is part of the Network Processing computer software component (CSC) of the Automatic Processing Computer Software Configuration Item (CSCI) and is identified as follows:

Title:	Global Association Subsystem
Abbreviation:	GA

SCOPE

This manual includes instructions for setting up the software, using its features, and basic troubleshooting. It does not describe the software's design. This topic is described in sources cited in "Related Information."

AUDIENCE

This document is intended for the first-time or occasional user of the software. However, more experienced users may find certain sections useful as a reference.

RELATED INFORMATION

The following documents complement this document:

- *Global Association (GA) Subsystem* [IDC7.1.4]
- *User Manual for the Global Association Subsystem* [LeB96]
- *IDC Processing of Seismic, Hydroacoustic and Infrasonic Data* [IDC5.2.1]

- *Database Schema, Revision 2* [IDC5.1.1Rev2]
- *Configuration of PIDC Databases* [IDC5.1.3]

See "References" on page 97 for a list of documents that supplement this document. The following UNIX manual (man) pages apply to the existing GA software:

- *GA*
- *GAcons*
- *GAassoc*
- *GAconflict*
- *GA_DBI*
- *GAgrid*

USING THIS DOCUMENT

This document is part of the overall documentation architecture for the IDC. It is part of the Technical Instructions category, which provides guidance for installing, operating, and maintaining the IDC systems. This document is organized as follows:

- Chapter 1: Introduction

  This chapter provides an overview of the software's capabilities, development, and operating environment.

- Chapter 2: Operational Procedures

  This chapter describes how to use the software and includes detailed procedures for startup and shutdown, basic and advanced features, and maintenance.

- Chapter 3: Troubleshooting

  This chapter describes how to identify and correct common problems related to the software.

- Chapter 4: Installation Procedures

  This chapter describes first how to prepare for installing the software, then how to install the executable files, configuration data files, database elements, and Tuxedo files. It also describes how to initiate operation and how to validate the installation.

- References

  This section lists the sources cited in this document.

- Appendix: GAgrid User Manual

  This section is a user guide for the interactive *GAgrid* application.

- Glossary

  This section defines the terms, abbreviations, and acronyms used in this document.

- Index

  This section lists topics and features provided in this document along with page numbers for reference.

Conventions

This document uses a variety of conventions, which are described in the following tables. Table I shows the conventions for data flow diagrams. Table II lists typographical conventions.

TABLE I: DATA FLOW SYMBOLS

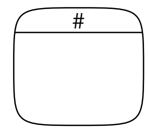
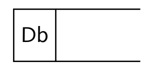
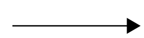
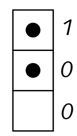
Description	Symbol
process	
data store Db = database	
data flow	
The vertical column of boxes represents an event hypothesis. The individual boxes represent arrivals. The numbers next to the boxes indicate the arrival's rank. The dot inside a box indicates the arrival is or has been associated with more than one event hypothesis.	

TABLE II: TYPOGRAPHICAL CONVENTIONS

Element	Font	Example
database table	**bold**	**arrival**
database table and attribute, when written in the dot notation		**arrival.***iphase*
database attributes	*italics*	*iphase*
processes, software units, and libraries		*GAgrid*
user-defined arguments and variables used in parameter (par) files or program command lines		*sigma_slowness*
titles of documents		*Configuration of PIDC Databases*
computer code and output	`courier`	`exit-ga-error=14`
filenames, directories, and websites		`/app_config/automatic`
text that should be typed exactly as shown		`edit-filter-dialog`

Chapter 1: Introduction

This chapter provides a general description of the software and includes the following topics:

- Software Overview
- Status of Development
- Functionality
- Inventory
- Environment and States of Operation

Chapter 1: Introduction

SOFTWARE OVERVIEW

Figure 1 shows the logical organization of the IDC software. GA is a subsystem in the Network Processing CSC of the Automatic Processing CSCI. Figure 2 shows the relationship of GA to other components of the Automatic Processing CSCI. This figure indicates that GA fulfills the following role: It associates the detections produced by Station Processing to form event hypotheses. Detections from the network of stations are grouped together into association sets, which define distinct events. These events are located, and their magnitudes are estimated. Figure 2 shows the dynamics of the processing, but does not show the static tables (**site**, **siteaux**, **affiliation**, and so on) or waveform data files used by the processes. Station Processing includes the programs *DFX*, which performs detection and feature extraction, *StaPro*, which performs initial phase identification, and *HAE* (Hydroacoustic Azimuth Estimation), which estimates the azimuth for multi-site hydroacoustic stations.

STATUS OF DEVELOPMENT

GA, first used operationally in March 1996, is an element of the Prototype International Data Centre (PIDC) at the Center for Monitoring Research (CMR) in Arlington, Virginia, U.S.A. and at the International Data Centre of the Comprehensive Nuclear-Test-Ban Treaty Organization (CTBTO IDC) in Vienna, Austria.

The first version processed only seismic data and ran in the first automatic processing pipeline along with its predecessor, Expert System for Association and Location (*ESAL*) [Bac93]. This processing configuration was used during the first year of the Group of Scientific Experts Third Technical Test (GSETT-3). Several upgrades of the subsystem have been made [LeB96]; the first to handle auxiliary seismic data in the later automatic pipelines and then to adapt for hydroacoustic [LeB96] and infrasonic data types [Kat98]. The subsystem architecture, however, has remained constant as GA has been extended to new data types.

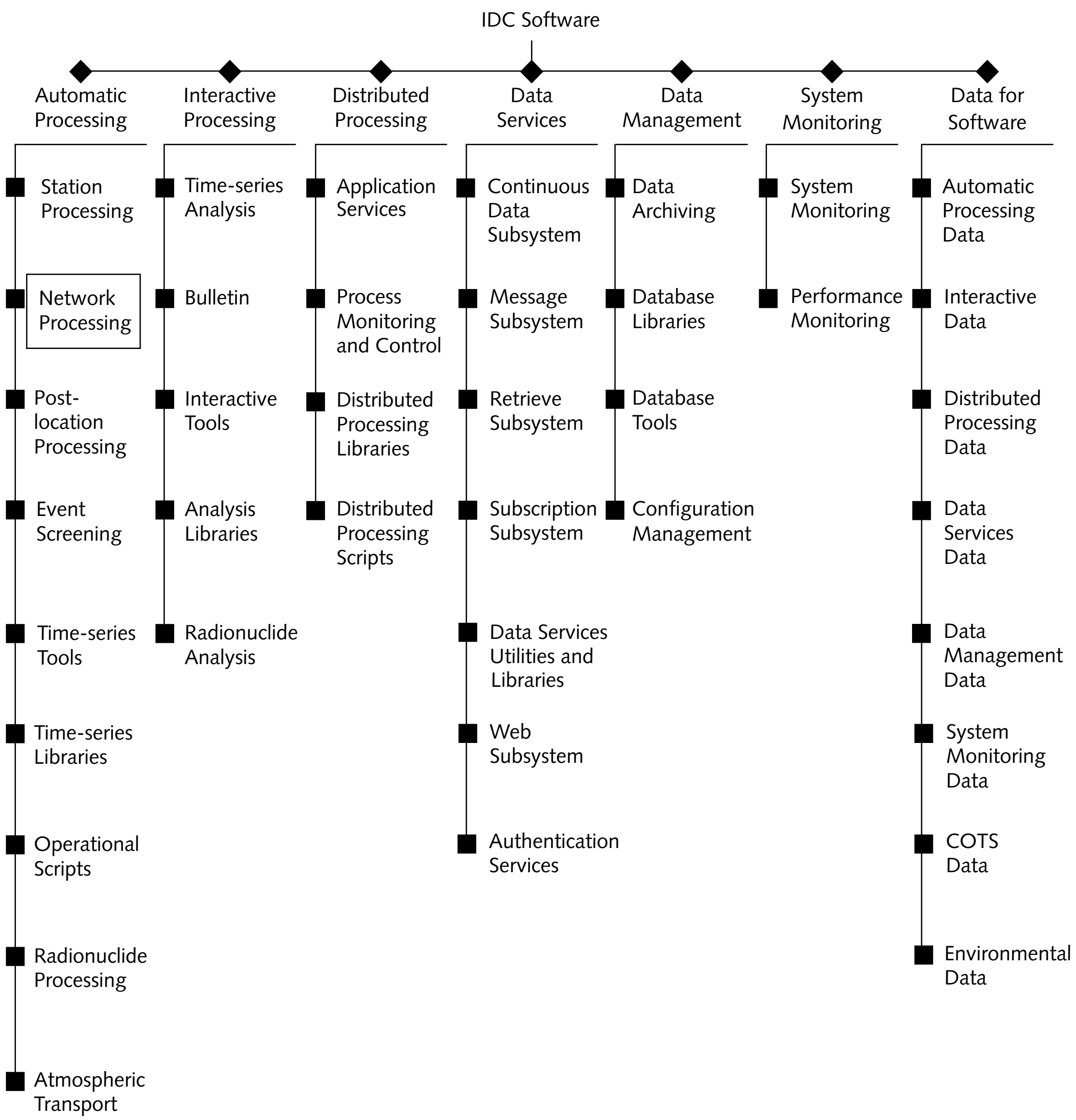


FIGURE 1. IDC SOFTWARE CONFIGURATION HIERARCHY

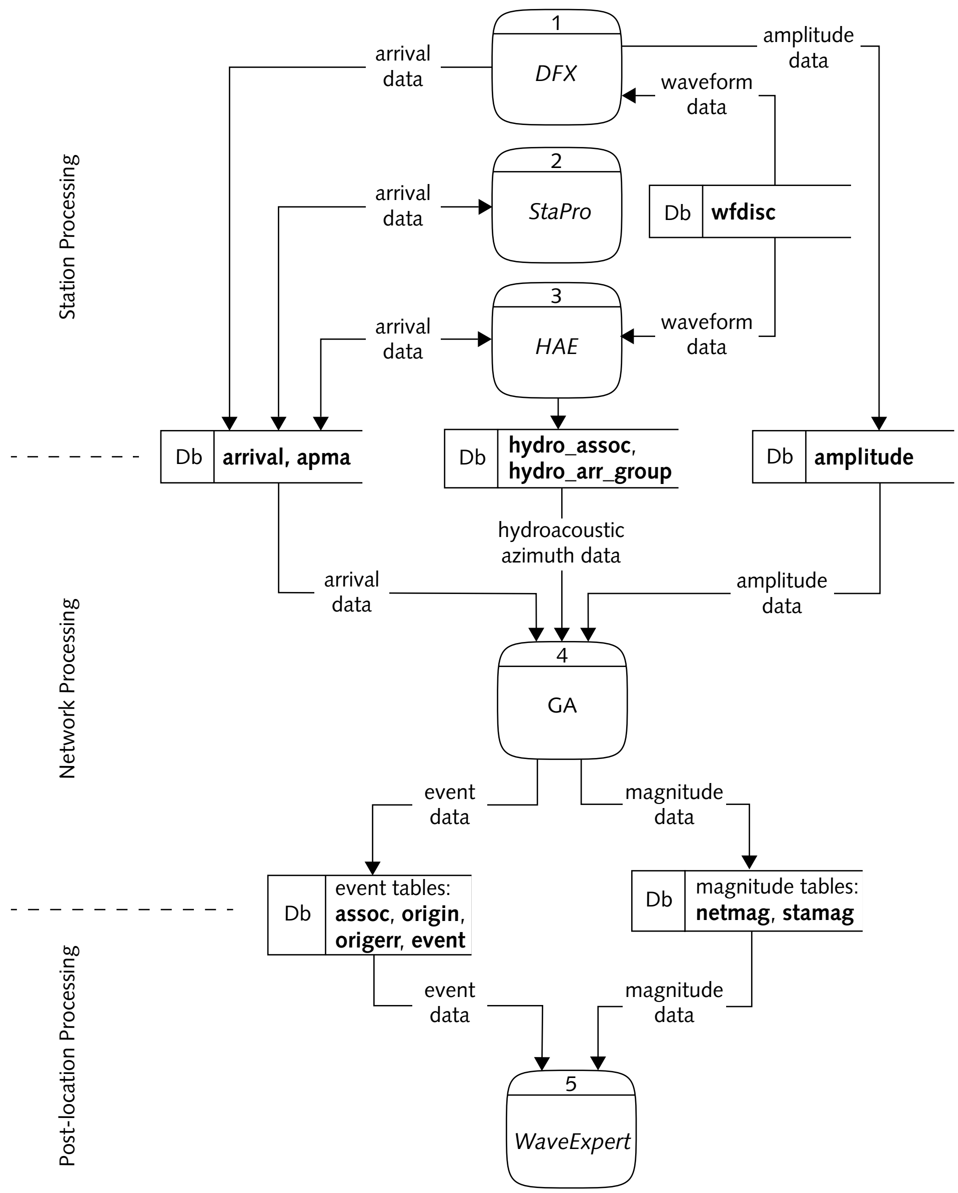


FIGURE 2. RELATIONSHIP OF GA TO OTHER SOFTWARE UNITS OF THE AUTOMATIC PROCESSING CSCI

FUNCTIONALITY

GA is the process in the automatic processing pipeline that forms event hypotheses. GA reads arrival and amplitude data for a time interval and forms sets of consistent associations using an exhaustive search algorithm. These association sets define the events, which then are located and have their magnitude estimated. The locations are estimated using a standard locator library, and the magnitudes are evaluated using a standard magnitude library.

GA's components (five programs and one library) are identified as follows:

- *GAassoc*
- *GAconflict*
- *GA_DBI*
- *GAcons*
- *GAgrid*
- *libGA*

GAassoc, *GAconflict*, and *GA_DBI* are used in the automatic processing pipelines. *GAcons* is a utility program used to generate two grid files used by the processing pipeline components, and *GAgrid* is a graphical user interface (GUI) program used for visualizing one of the grid files.

Two grid files generated by the *GAcons* utility are used in the creation of the automatic bulletin. The first grid file, referred to as the "Propagation Knowledge-base (PKB) grid file," contains the travel-time and propagation characteristics for all stations in the network and all grid cells in the grid file. The second grid file, the Static grid file, contains all of the static information about the grid cells, such as location and size. The information contained inside these grid files is subject to update with changing networks, so care must be taken to upgrade the grid files every time a change takes place in any of the networks, such as the addition of a new station.

The input database tables **arrival**, **amplitude**, and **apma** contain features measured from the data using the Station Processing software *DFX*. GA writes the output tables **origin**, **origerr**, **assoc**, **event**, **netmag**, and **stamag** in the database account specified for the pipeline. These tables serve as input to the subsequent processes.

GA interfaces with several other components of the IDC processing system. The primary interfaces are:

- the Relational Database Management System (RDBMS)
- the locator library, which is invoked to compute the **assoc**, **origin**, and **origerr** records produced by the locator module of GA
- the travel-time handling facility, which is invoked when building the PKB grid file and also in both GA prediction modules

Features and Capabilities

GA was initially designed to create an automatic seismic bulletin, but it has been upgraded to identify events that include arrivals from hydroacoustic and infrasonic stations. A flexible number of networks for each of the technologies can be specified, and events can be created that belong to either one of the technologies or to a mixture of the technologies.

GA has the capacity to process data either independently of previous processing pipelines or in conjunction with previously processed data. The former configuration is used in the automatic processing of the SEL1 (Standard Event List) pipeline. The latter configuration is used in the automatic processing of the SEL2 and SEL3 pipelines.

A large number of user parameters support a highly-flexible configuration. GA has been installed at several sites and the configuration has been specifically tailored to the various needs at these sites.

Performance Characteristics

GA's performance can be characterized by its processing time and by the quality of its bulletin. GA normally easily runs faster than real time. Of the three pipeline GA processes, the bulk of the time is taken by *GAassoc*. The processing time is a function of several factors, most notably the number of detections with defining phases (available to define an event), the number of events, and the presence of overlapping large events. In only a handful of time intervals in the last two years has

GAassoc taken more than half of the available time; all but one of these occurred during a swarm sequence on 16 November 2000. Some characteristics of the 10 days preceding and including that day are presented below.

- The principal inputs to automatic association processing are the detections in the **arrival** table. To illustrate a typical rate of input to GA, the average daily rate of automatic detection at the IDC at the primary seismic stations for the period of 7 November through 16 November was 100 detections per day per station with 28 active primary seismic stations. No problems were observed in the automatic network processing during that time period, although some stations were off-line (for example, WRA). This time period has a large number of events from a swarm associated with a large event in the region of Papua New Guinea on 16 November 2000. The average daily rate varies over the different stations from a low of 9 per day to a high of 234 per day.
- Table 1 shows the number of events formed by GA in the three automatic processing pipelines at the IDC for the same 10-day period shown in Table 1. 7 November to 15 November have typical levels of seismic activity, but 16 November is characterized by an unusually high level of seismic activity in the general area of Papua New Guinea.

TABLE 1: AVERAGE DAILY EVENT RATES IN AUTOMATIC BULLETINS

Date	Total Number of Arrivals	Number of Associated Detections (SEL3)	Number of Events		
			SEL1	SEL2	SEL3
7 – 15 November 2000	6,643	656	94	73	73
16 November 2000	16,455	4,874	434	375	369

- *GAassoc* requires the most processing time of the pipeline components. The average *GAassoc* processing time for the SEL3 bulletin was 34 seconds on 14 November, a day with an average number of detections, and increased to 102 seconds on 16 November, using a Sun Ultra Enterprise server. The processing time was a small fraction of real time, in all cases, even when there was a large swarm of events. These are typical processing times in IDC operations. To estimate upper bounds on these processing times, simulations have been run using synthetic detection data that characterize some of the most stressful conditions observed during PIDC operations. The simulations used the planned full IMS network and showed that the software is able to handle all conditions that were examined.

- The quality of the GA bulletin can best be characterized by two numbers: the percentage of events successfully formed automatically and confirmed by the analyst (detected event rate), and the false-alarm rate, which is the number of events that were formed by the automatic system and not confirmed by the analyst. False alarms may, in some cases, actually be real events, but may not meet strict event definition criteria. A survey of unit test results indicates that detected event rates range from about 80 percent to 90 percent whereas false-alarm rates vary from about 95 percent to 105 percent. The exact values of these rates depend on the definition used for a detected event.

Related Tools

The script *Clean_tables-and-MarkState.sql* is delivered along with the GA executables, and its execution is required for GA to function properly. The script deletes all entries from the temporary tables **origin**, **origerr**, and **assoc** and updates the **interval** table to reflect new states of GA processing in preparation for running a new instance of *GAassoc*. The script is called a "start-hook" of the *GAassoc* process, which means that it is run before every new instance of *GAassoc*. The mechanism for start-hook and stop-hook processing is part of the *tuxshell* processing setup. Because *GAassoc* writes to empty tables, *GAconflict* can read the entire contents of these tables and does not have to query for specific records.

INVENTORY

GA uses three executables, *GA_DBI*, *GAassoc*, and *GAconflict*, in each automatic processing pipeline to process the detections and assemble them into event hypotheses. The utility program, *GAcons*, is used to construct the two grid files necessary to perform the automatic processing in *GAassoc* and *GAconflict*.

The database tables used as input by GA can be categorized as static or dynamic. The static database tables **site**, **siteaux**, and **affiliation** are available in the STATIC database account. The dynamic tables **arrival**, **amplitude**, **apma**, **hydro_assoc** and **hydro_arr_group** are written by Station Processing and are available in the IDCX database account.

The output from GA is written to database tables that contain the automatic bulletin information. These tables include **assoc**, **origin**, **origerr**, **event**, **netmag**, and **stamag**, and can be either temporary or final. *GAassoc* writes its output to a set of temporary tables: **assoc_temp_ga**, **origin_temp_ga**, and **origerr_temp_ga**. *GAconflict* reads these tables and writes its output to **assoc**, **origin**, **origerr**, **event**, **netmag**, and **stamag**. One set of temporary tables and one set of final tables exists in each of the SEL1, SEL2, and SEL3 accounts. In addition, *GA_DBI* writes to the **ga_tag** table, which is read by *GAassoc* and *GAconflict*.

A number of files are necessary for the software to operate. They include the configuration files in <*dir*>`/earth_specs/TT` (the directory <*dir*> is set using a UNIX variable). The velocity model specification file (VMSF) in directory <*dir*>`/earth_specs/TT/vmsf` is set through the use of parameter *vmodel_spec_file*. The VMSF file specifies which directories and files of the travel-time configuration directory <*dir*>`/earth_specs/TT` are accessed. Similarly, the magnitude specification file (MDF) in directory <*dir*>`/earth_specs/MAG/mdf` and the transmission loss specification file (TLSF) in directory <*dir*>`/earth_specs/MAG/tlsf` are set through the use of parameters *magnitude_spec_file* and *transmission_loss_spec_file*. These files specify the parts of directory <*dir*>`/earth_specs/MAG` that are used by the software. The file <*dir*>`/earth_specs/MAG/atten/slowamp.P` and all of the files in directory <*dir*>`/earth_specs/GA` are also needed to run the software.

ENVIRONMENT AND STATES OF OPERATION

Software Environment

GA is written in ANSI C (American National Standards Institute) and can be run on any hardware supporting the standard. The software was developed and tested uniquely on Sun workstations running the Solaris operating system. The recommended configuration for the software is 128 MB of RAM (Random Access Memory) on a Sun workstation, an UltraSPARC processor, and 2 or 3 GB of disk space.

GA was designed to work in conjunction with an ORACLE RDBMS. No option exists to use the software with an alternative system, such as flat files.

GA was designed to be used primarily in an automatic pipeline context where the execution sequence of the GA programs (*GA_DBI*, *GAassoc*, and *GAconflict*) is controlled by the Distributed Application Control System (DACS). The DACS uses the commercial-off-the-shelf software (COTS) Tuxedo.

Normal Operational State

In normal operations, *GAcons* is invoked manually to build the two grid files used by the pipeline executables, and the pipeline executables *GA_DBI*, *GAassoc*, and *GAconflict* are controlled by the DACS.

Contingencies/Alternate States of Operation

The pipeline executables, *GA_DBI*, *GAassoc*, and *GAconflict*, may be run outside of the DACS context with appropriate UNIX scripts, which invoke the executables in the proper order. This setup was used during the development and tuning phases of the software when it was desirable to run regression tests on a fixed data set. A typical test setup is to run the processing suite on a data set containing a few days worth of input data.

Chapter 2: Operational Procedures

This chapter provides instructions for using the software and includes the following topics:

Chapter 2: Operational Procedures

SOFTWARE STARTUP

The GA software is used in the context of automatic processing pipelines. The standard method of starting the programs is through the command line, with the command line parameters appropriate for the given pipeline. The software is delivered with configuration files for three pipelines (SEL1, SEL2, and SEL3). Parameters are adjusted appropriately for each of these pipelines, which includes using the results from earlier pipelines as input to the current pipeline.

In the automatic processing pipeline configuration used at the IDC, GA is called by the *tuxshell* application, which provides the services appropriate within a given pipeline. The various services provided by DACS in the three automatic processing pipelines are detailed in [IDC6.5.2Rev0.1]. The *tuxshell* application server runs permanently both on the host server targeted for the application and on its backup server. The *GA_DBI*, *GAassoc*, and *GAconflict* applications must run sequentially; one application must finish executing before the next one is called. In addition, the intervals of data must be processed in chronological order. The three applications each run once, in sequence, on a given time-interval of data before proceeding to the next interval. This control is handled by the Tuxedo-based DACS, which uses compound processing to ensure that *GA_DBI*, *GAassoc*, and *GAconflict* all successfully complete processing for the interval. Specifically, after each service request is made to *tuxshell* for one of the applications, the DACS calls the application that is next in the sequence by initiating the next request for services to the appropriate *tuxshell* instance.

In normal IDC operations, use the *tuxpad* GUI program to initiate the operation of the GA applications that are part of the automatic processing pipelines. Select the services corresponding to the pipeline(s) to be initiated, that is, SEL1, SEL2, and/or SEL3. The exact steps necessary to initialize the services are described in [IDC6.2.1].

GA can also be called as stand-alone executable binaries using scripts. This option is useful during development and for conducting tuning and special studies where regression analysis is important. The following shell script executes GA for the time period between `starttime` and `stoptime`:

```
#!/bin/csh
#
# This script executes the sequence of GA processes in the
# prescribed order of GA_DBI, GAassoc and GAconflict, with
# an additional call to a script (DelTables.sql) that
# clears the temporary tables.
# The preclear.sql script clears the output tables
# before starting the processing.
#
#
set starttime = 896054400
set stoptime = 896572800
set inc = 1200
set t1 = $starttime
set command = (sqlplus ga/ga@database @preclear.sql hyaz)
echo $command ;
echo === TIME ===  'date';
$command
loop:
@ t2           = $t1 + $inc
set command = (sqlplus ga/ga@database @DelTables.sql\
               origin_temp_ga origerr_temp_ga assoc_temp_ga)
echo $command ; echo === TIME ===  'date'; $command
set command = (GA_DBI par=GA_DBI.par start_time=$t1\
               end_time=$t2 global_verbose=1)
echo $command ; echo === TIME ===  'date'; $command
```

```
set command = (GAassoc par=GAassoc.par start_time=$t1\
               end_time=$t2 global_verbose=1)
echo $command ; echo === TIME ===  'date'; $command
set command = (GAconflict par=GAconflict.par start_time=$t1\
               end_time=$t2 global_verbose=1)
echo $command ; echo === TIME ===  'date'; $command

if ($t2 >= $stoptime) exit
  @ t1          = $t1 + $inc
  goto loop
```

The most common problem encountered in running stand-alone is failure to properly configure to use the intended database tables. See "Chapter 3: Troubleshooting" on page 55 for more information.

SOFTWARE SHUTDOWN

Use the *tuxpad* shutdown facilities to shut down the GA services started through the DACS Tuxedo. The steps necessary to shut down the software are described in more detail in [IDC6.5.2Rev0.1]. The shutdown can be either selective or complete. A selective shutdown stops only a restricted number of the GA services. A total shutdown stops all Tuxedo services.

Use the following command in *tmadmin* to selectively shut down a specific GA service:

```
> shutdown -i SRVID
```

SRVID is the logical reference to a *tuxshell* server for a specific service of GA. For instance `GA-sel1:tuxshell` is the server for the *GA_DBI* application, which is the first application in the GA processing suite in the SEL1 pipeline.

Use the following *tmadmin* command to shut down the whole Tuxedo application:

```
> shutdown -y
```

When running a stand-alone test, the process can be terminated with standard UNIX kill commands. All processes initiated by the script must be terminated.

BASIC PROCEDURES

GA includes both configuration software (*GAcons* and *GAgrid*) and pipeline processing software (*GA_DBI*, *GAassoc*, and *GAconflict*). Use *GAcons* and *GAgrid* only when needed to generate or examine the grid files. Call these programs through the use of a command line interface:

```
GAcons par=GAcons.par

GAgrid par=GAgrid.par
```

After the grid files have been created, the pipeline processing GA software can be run as part of IDC processing. The different executable binaries are called on the UNIX command line with command line parameters to set up the specific input and output filenames, table names, and numerical and character string parameters. The script shown in the section on "Software Startup" on page 12 provides examples of command line calls to *GA_DBI*, *GAassoc*, and *GAconflict*.

To set up stand-alone runs of the pipeline processes, create the database tables used by GA. These tables include the standard **origin**, **origerr**, **assoc**, **event**, **stamag**, and **netmag** tables as well as the temporary tables **origin_ga_temp**, **origerr_ga_temp**, and **assoc_ga_temp**. When running GA stand-alone, these tables are usually given a different name than the default names mentioned in the previous sentence to segregate results and avoid conflicts between different processing runs of the same input data. The static tables **site**, **siteaux**, and **affiliation** also must be present in the database, as well as the input tables **arrival**, **apma**, and **amplitude**.

To install GA at a new location for the purpose of standard operations, the database tables in the IDCX and SEL*n* accounts must be present. The parameters that set table names for the software have default table names that match the account table names; customizing is not necessary. The travel-time tables, magnitude tables, and grid files are all located in standard locations set by system-wide parameters.

Obtaining Help

Help in using this software is provided in this user's manual, through the man pages (see "Related Information" on page ii), and in the document *Processing of Seismic, Hydroacoustic, and Infrasonic Data* [IDC5.2.1].

ADVANCED PROCEDURES

This section provides instructions for using some of the software's advanced features. The first two sections describe verbosity parameters that can be used to debug GA or investigate its behavior, and provide details on how to use the Debug module to investigate the causes for a missed event. The largest portion of this section shows how the user parameters influence the GA algorithmic behavior.

Setting Verbosity Levels

A large number of parameters are available to set the verbosity level in GA. They are *global_verbose*, *assoc_verbose*, *ev_verbose*, *cr_verbose*, *loc_verbose*, and *debug_verbose*. As indicated by their name, each of the parameters is targeted to provide information about a specific area of processing. Each of these parameters can take a value from 0 to 4. The higher the value, the more detailed the provided information.

Set the *global_verbose* parameter to control the verbosity level for the overall GA processing. A value of 4 (usually avoided) produces a large volume of output, including the complete linked list of preliminary event hypotheses at various stages of processing.

Set the *assoc_verbose* parameter to control the verbosity level for the association loop portion of the processing. Use a level of verbosity higher than the default level for this parameter to get more information about the details of the association process such as the initial steps of event formation. For example, the selection of a driver arrival and the association of corroborating arrivals to the driver arrival may be followed in detail.

Set the *ev_verbose* parameter to control the verbosity level for the event confirmation portion of the processing. Use a level of verbosity higher than the default level for this parameter to get more information on event confirmation, such as a message indicating the *orid* number of event hypotheses that fail event confirmation, as well the name of the event confirmation test that fails.

Set the *cr_verbose* parameter to control the verbosity level for the conflict resolution portion of the processing. Use a level of verbosity higher than the default level for this parameter to get more information about the conflict resolution process. For example, a medium verbosity level displays tables showing the values of the various membership function weights and the ranking of associations based on these values at each iteration of the conflict resolution process.

Set the *loc_verbose* parameter to control the verbosity level for the location portion of the processing. Use a level of verbosity higher than the default level for this parameter to get more information about the location process. For example, a higher verbosity level provides information about each iteration of the location algorithm every time that a location is performed.

Set the *debug_verbose* parameter to control the verbosity level for the `GA_debug` module of GA, which is useful for tracking the formation and evolution of an event hypothesis specified in an auxiliary database table. The auxiliary database table is specified by parameter *debug_assoc_table*. More information on this module is provided in the next section.

Using GA_debug Module

Use the `GA_debug` module to track event hypotheses through the GA processing sequence. It can be particularly useful in determining the cause of a missed event. The module is inserted at various places within the processing sequence. To specify the event to be traced through the processing sequence, list its associations in a database table with the structure of the **assoc** table. Specify the name of the database table using parameter *debug_assoc_table*. In addition, use the parameter *debug_where* to specify the SQL WHERE clause that selects the event in the *debug_assoc_table* that is compared to the GA event hypotheses. Use the parameter *debug_threshold* to specify the number of matching associations that define a GA event hypothesis as matching an event in the *debug_assoc_table*. Only the event hypotheses that satisfy this criterion are reported in the `GA_debug` output. Adjust the *debug_verbose* parameter to determine the level of information reported when a matching event is found. When set to a value of 0, no output is produced. With a value of 1, the module reports the number of distinct event hypotheses in the *debug_assoc_table*, the number of event hypotheses in the *debug_assoc_table* with at least *debug_threshold arid*s, and the number of event hypotheses in the *debug_assoc_table* that match at least one event hypothesis. With a value of 2, for every event in the *debug_assoc_table* with at least *debug_threshold arid*s, the module reports its *orid*, the number of associations, and the number of matching event hypotheses. With a value of 3, for every event in the *debug_assoc_table* with at least *debug_threshold arid*s, the module reports its *orid*, the number of associations, and for every matching event hypothesis, and the matching *arid*s separated from the non-matching *arid*s by a star.

Setting User Parameters

GA is used normally as a stable component of the automatic processing pipeline, and the parameter values are not usually changed in the configuration files delivered with the software. User parameters are used to specify site-specific configuration, such as the locations of various files, and to specify the details of some of the algorithms implemented in GA. In this section, the links between the parameter names, as they are explained in the UNIX man pages, and the description of the algorithms provided in [IDC5.2.1] are clarified. This allows one to understand the algorithmic effects of modifying the parameter values. The sections that follow correspond to the processes that compose the different GA programs, as shown in dataflow diagrams in [IDC7.1.4]. Table 2 shows a mapping between the activities described in the following sections and the GA applications.

TABLE 2: MAPPING BETWEEN APPLICATIONS AND ACTIVITIES

Application	Activities
GAcons	Building Static Grid Building Grid Files
GA_DBI	Tagging Arrivals for Subsequent Processing
GAassoc	Extracting Arrival Lists Associating Arrivals Extracting Large Events Eliminating Redundant Event Hypotheses Locating and Confirming Preliminary Event Hypotheses Resolving Conflicts
GAconflict	Extracting Arrival Lists Eliminating Redundant Event Hypotheses Locating and Confirming Preliminary Event Hypotheses Resolving Conflicts Predicting Non-defining Phases Predicting Defining Phases Performing Seismological Checks

Some of these activities may belong to more than one of the applications. For instance "Resolving Conflicts" on page 41 is performed within both *GAassoc* and *GAconflict*.

Two categories of parameters can be identified: those that affect the installation (for example, database table names, filenames) and those that control the algorithms. The most interesting and significant of the latter category are described in this section. Table 3 lists the parameters used to configure each activity, the applications that the activity belongs to, and the parameter files that contain the listed parameters.

TABLE 3: PARAMETER MAPPING

Application	Activity	Parameter	Par file
GAcons	Building Static Grid	*grid_spacing*	`GAcons.par`
		event_file	
		depth_points	
		depth_widths	
		min_num_events_ per_10sq_deg	
		blockage_spec_dir	
		vmodel_spec_file	
		hydro_phases	
		infra_phases	
	Building Grid Files	*phases*	
GA_DBI	Tagging Arrivals for Subsequent Processing	*net*	`GAsel{1,2,3}.par`
		net_aux	
		hydro_snr_thresh	`GA_DBI_sel{1,2,3}.par`
		hydro_max_H_per_hour	

TABLE 3: PARAMETER MAPPING (CONTINUED)

Application	Activity	Parameter	Par file
GAassoc *GAconflict*	Extracting Arrival Lists	*in-site-table*	`GAassoc_sel{1,2,3}.par` `GAconflict_sel{1,2,3}.par`
		in-affiliation-table	
		in-arrival-table	
		in-amplitude-table	
		in-assoc-table	
		in-origin-table	
		in-origerr-table	
		out-origin-table	
		out-assoc-table	
		out-origerr-table	
		pre-origin-table	
		pre-assoc-table	
		pre-origerr-table	
	Eliminating Redundant Event Hypotheses	*count_limit*	`GA.par`
	Locating and Confirming Preliminary Event Hypotheses	*loc_fix_depth*	
		chi_outlier	
		max_smajax	
		req_num_of_defining_detections	
		max_obs_net_prob	
		residual_over_sigma_max	

TABLE 3: PARAMETER MAPPING (CONTINUED)

Application	Activity	Parameter	Par file
GAassoc *GAconflict*	Resolving Conflicts	*cluster_min_ndef*	`GA.par`
		cluster_min_pct_overlap	
		master_tradeoff_weight	
		event_likelihood_weight	
		dissolved_event_weight	
		ndef_no_confidence_bound	
		ndef_high_ confidence_bound	
		ndef_weight	
		smajax_no_ confidence_bound	
		smajax_high_ confidence_bound	
		smajax_weight	
		dnear_no_ confidence_bound	
		dnear_high_ confidence_bound	
		dnear_weight	
		probdet_no_ confidence_bound	
		probdet_high_ confidence_bound	
		probdet_weight	
		ndef_not_likely_to_ dissolve_event	
		ndef_which_will_ dissolve_event	

TABLE 3: PARAMETER MAPPING (CONTINUED)

Application	Activity	Parameter	Par file
GAassoc	Associating Arrivals	*start_time*	command line
		end_time	
		input_path	`GA.par`
		input_file	
		num_first_sta	
		driver_phases	
		origin_time_limit	
		end_time_seismic	
		end_time_hydroacoustic	
		end_time_infrasonic	
		sigma_slowness	
		forward_transformation_list	
		primary_required_for_secondary	
		regional_S_phases	
		chi_limit	
		hydro_corroborate_hydro_only	
		infra_corroborate_infra_only	
		primary_time_weight	
		secondary_time_weight	
		hydro_time_weight	
		infra_time_weight	
		array_azimuth_weight	

TABLE 3: PARAMETER MAPPING (CONTINUED)

Application	Activity	Parameter	Par file
GAassoc	Associating Arrivals	*3comp_azimuth_weight*	`GA.par`
		hydro_azimuth_weight	
		infra_azimuth_weight	
		array_slow_weight	
		3comp_slow_weight	
		hydro_slow_weight	
		infra_slow_weight	
		weight_threshold	
		ar_qual_alpha	
		ar_qual_beta	
		ar_qual_thresh	
		ar_qual_alpha_hydro	
		ar_qual_beta_hydro	
		ar_qual_alpha_infra	
		ar_qual_beta_infra	
		dist_depth_range_file	
	Extracting Large Events	*min_ndef_extract*	default value

TABLE 3: PARAMETER MAPPING (CONTINUED)

Application	Activity	Parameter	Par file
GAconflict	Predicting Non-defining Phases	*predicted_phases*	`GA.par`
		chi_outlier	
	Predicting Defining Phases	*defining_phases*	
	Performing Seismo-logical Checks	*gridfile*	
		dist_depth_range_file	
		equivalence_list	
		chi_outlier	
		primary_required_ for_secondary	
		min_sta_magnitude_diff	
		max_uncertainty_ magnitude_diff	
		default_max_ magnitude_diff	
		max_magnitude_diff	
		max_ndef_reset_fix_depth	
		phase_ordering_list	
		max_reported_mag_sig	

One component of the GA subsystem, *GAgrid*, is an interactive program. By its nature, *GAgrid* is different from the other four GA programs and it is documented separately in “Appendix: GAgrid User Manual” on page A1.

The following two sections describe the activities listed for application *GAcons* in Table 2. The first section describes how the static grid is built. The second section relates how the propagation information is computed to construct the PKB grid file and describes the use of the two grid files.

Building Static Grid

This section describes the procedures for building the PKB grid file using *GAcons*. The procedure is described in four parts: surface cells, depth cells, hydroacoustic phases, and infrasonic phases. The first two parts describe how to build the grid used to perform the association process; the second two sections describe how to include the non-seismic phases in the grid.

Surface Cells

GAcons establishes grid points for the earth with a quasi-uniform spacing. The grid points are distributed along parallels. The spacing in latitude is uniform, and the number of latitude lines (*Nlat*) between 0 and 90 degrees is computed in equation (1):

$$\mathrm{Nlat} = \frac{\sqrt{2}}{\Delta}90^\circ \tag{1}$$

where Δ is the average grid spacing in degrees, which is set through the *grid_spacing* parameter. The square-root-of-two factor provides a large enough density in latitude to ensure complete coverage by overlapping cells. A point is always at the North and South Poles.

The points are spaced uniformly for each latitude line. The number of points, $N(\lambda)$, and the uniform spacing in degrees of longitude, $\delta(\lambda)$, between them are computed using equations (2) and (3):

$$N(\lambda) = 1 + \frac{(\cos\lambda)360^\circ}{\Delta} \tag{2}$$

$$\delta(\lambda) = \frac{360^\circ}{N(\lambda)} \tag{3}$$

where λ is the latitude. The grid is built starting with uniformly distributed points on the equator; one point at longitude=0. The next latitude line has a point placed at longitude $\delta(0)/2$, exactly half-way in longitude between two points at the equator. This process is repeated for each latitude line, where one point is placed exactly half-way between two points at the latitude directly below. At lower latitudes, this results in a quasi-quincunx pattern rather than a square grid. The radius of each surface cell is set to the latitude spacing, which provides a sufficiently large overlap to ensure complete coverage of the earth.

Depth Cells

The generation of depth cells in the grid is guided by a file containing events from a published global seismic bulletin (for example, the Preliminary Determination of Epicenters) recorded over many years as specified by the parameter *event_file*. This file lists one event per line formatted as:

<latitude> <longitude> <depth> <m_b>

The depths and thicknesses of the depth cells are specified by the parameters *depth_points* and *depth_widths*. Every surface location is included in the grid. The depth cells below the surface locations are included if the number of events in the cell exceeds the specified threshold event density from the seismicity bulletin (*min_num_events_per_10sq_deg*). The cell shape is a portion of a cone truncated at the lower and upper depth of the cell.

Hydroacoustic Phases

If there is a clear path from a hydroacoustic station to at least a few points within a surface cell, information about the propagation of hydroacoustic phases to that station is included in the grid cell record. The hydroacoustic blockage grid for that station determines whether or not a clear path exists from the station to the grid cell. Hydroacoustic phases propagate in the deep oceans, and little of their energy couples into the solid earth. The directory containing the blockage files is specified by the parameter *blockage_spec_dir*. Hydroacoustic propagation information is specified in two-dimensional (2-D) grids. The travel-time handling system, called through the *libloc* library, sets the appropriate values in the parameter input file specified by *vmodel_spec_file*. The directory of the 2-D grid files that specify the azimuth-dependent propagation files is set inside the velocity model specification file. The list of possible names for hydroacoustic phases is provided by the parameter *hydro_phases*.

Infrasonic Phases

For infrasonic phases, the surface cell consists of an atmospheric cell, with a top altitude set by the symbolic constant *UPPER_BOUND_ATMOSPHERE*, currently set to 200 km in `libGA.h`. The list of possible names for infrasonic phases is provided by the parameter *infra_phases*.

Building Grid Files

After the static grid has been established, the propagation information from stations within the networks to all the grid cells is computed and saved in the PKB grid file. The static grid information is also saved in a separate file, called the Static grid file.

The PKB grid file is used by *GAassoc* exclusively. *GAassoc* reads the precomputed propagation information contained within the file and uses it to build event hypotheses. The information contained within the file consists of the geographic and size information for each of the cells and the travel-time and slowness information for all phases specified by parameter *phases*, which may include seismic, hydroacoustic, and infrasonic phases.

The Static grid file is used exclusively by *GAconflict* and more specifically by the function `GA_check_depth()`, which performs one of the consistency checks within *GAconflict*. The file contains the geographic and size information for each of the grid cells. This information is also contained within the PKB grid file, but its size is smaller, so the entire content can be loaded into memory for processing by `GA_check_depth()`.

Tagging Arrivals for Subsequent Processing

GA_DBI is the first program to run in each of the three automatic processing pipelines (SEL1, SEL2 and SEL3). It has two major functions: The first function is to tag arrivals from auxiliary stations; the second function is to tag hydroacoustic arrivals to avoid processing overload.

Auxiliary Station Tagging

GA_DBI's first major task is to tag the arrivals from auxiliary stations as `wc_restricted`, `probdet_restricted`, and `requested`. These tags are subsequently used by the *GAassoc*/*GAconflict* programs to eliminate the auxiliary stations from the weighted-count test for event confirmation and from the probability-of-detection test. The GA parameters relevant to this tagging are *net* and *net_aux*. Arrivals belonging to the network(s) described by the parameter *net_aux* are tagged, whereas the arrivals belonging only to the *net* network(s) are not tagged.

Hydroacoustic Station Tagging

GA_DBI's second task is to tag hydroacoustic arrivals as `driver_restricted` under circumstances defined by the two parameters *hydro_snr_thresh* and *hydro_max_H_per_hour*. This ensures that the GA subsystem does not saturate due to a large number of H phases in the set of arrivals. All arrivals whose signal-to-noise ratio (snr) is lower than *hydro_snr_thresh* are tagged as `driver_restricted`. In addition, if there are more than *hydro_max_H_per_hour* arrivals above the *hydro_snr_thresh* in the hour before the end time of the interval to be processed by *GA_DBI*, the threshold is adjusted so that no more than *hydro_max_H_per_hour* arrivals can ever be used as drivers in *GAassoc*.

Extracting Arrival Lists

The *Extract Arrival List* process is common to *GAassoc* and *GAconflict* and builds the list of arrivals for the interval to be processed by these GA programs. *GAassoc* requires the following input:

- **site** and **affiliation** database tables set by the parameters *in-site-table* and *in-affiliation-table*
- **arrival** and **amplitude** database tables specified by the parameters *in-arrival-table* and *in-amplitude-table*
- start time and end time set by the parameters *start_time* and *end_time*

GAconflict requires the following input:

- temporary tables produced by *GAassoc* and specified by the parameters *in-assoc-table*, *in-origin-table*, and *in-origerr-table*
- current output tables specified by the parameters *out-origin-table*, *out-assoc-table*, and *out-origerr-table*
- previous output tables specified by the parameters *pre-origin-table*, *pre-assoc-table*, and *pre-origerr-table*

Associating Arrivals

The initial association process consists of several steps described in this section. This process is used only by the *GAassoc* program and makes extensive use of the grid generated by *GAcons*. No other process within GA generates event hypotheses. The PKB grid file is read from the directory indicated by the parameter *input_path* and the filename set by the parameter *input_file*.

Identifying Driver Arrivals

The first task for each grid cell is to identify driver arrivals. A driver is an arrival at one of a limited set of stations in the network that can record the earliest arrival for an event in the given grid cell. The set of stations is determined by *GAcons* and is stored in the grid file. The maximum number of stations that are considered for any grid cell is specified by the parameter *num_first_sta.* The parameter *driver_phases* lists acceptable values for the **arrival**.*iphase* of a driver arrival. *GAassoc* processes arrivals in time-steps. To ensure that all corroborating arrivals of interest are available in the time-step, arrivals are loaded according to their technology. For seismic arrivals, they are loaded from *start_time–origin_time_limit* to *start_time+end_time_seismic*. For hydroacoustic arrivals, they are loaded from *start_time–origin_time_limit* to *start_time+end_time_hydroacoustic*. Finally, for infrasonic arrivals, they are loaded from *start_time–origin_time_limit* to *start_time+end_time_infrasonic.* Arrivals with arrival times before *end_time* are considered as potential drivers. The driver must satisfy constraints on its slowness vector. After a driver is identified, a hypothetical event origin time, magnitude, and location are estimated.

The length of the vector difference between the observed slowness vector of the driver arrival and the predicted slowness vector must be less than a threshold (Figure 3). The threshold is the sum of the standard deviation of the slowness for the arrival (from **arrival.***delslo*) multiplied by a user-specified factor (σ) and the maximum difference between the slowness vector to the center of the cell and to any other point in the cell (Δ cell).

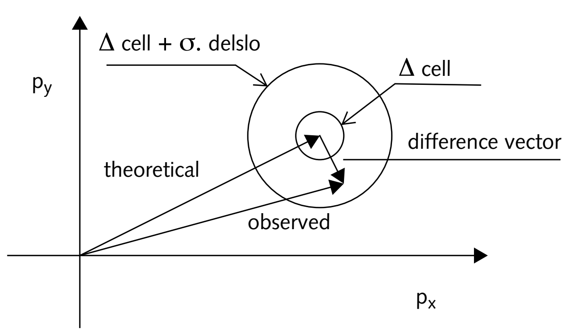


FIGURE 3. DIFFERENCE BETWEEN THEORETICAL AND OBSERVED SLOWNESS VECTORS

The user-parameter *sigma_slowness* specifies σ. The maximum slowness vector difference (Δ cell) for a specific station, phase, and cell is computed by *GAcons* and stored in the grid file.

Hydroacoustic arrivals are used as driver arrivals only if they might emanate from an underwater explosion (H phases in **arrival**.*iphase)*. Hydroacoustic driver arrivals are limited to prevent overloading the number of preliminary event hypotheses, due to the absence of azimuthal information at some of the hydroacoustic stations.

For infrasonic phases, azimuth information is available for all identified I phases, and all infrasonic arrivals with azimuth information are potential driver arrivals.

Searching for Corroborating Arrivals

The next step in the association process is to search for arrivals that are consistent with the driver. These arrivals are called corroborating arrivals. This is done by examining arrivals at all stations at times later than the driver arrival, where the arrivals are restricted to phases in the parameter list *phases*. Alternative phase names to those determined by station processing for these arrivals are considered from a user-specified list (*forward_transformation_list*). An optional restriction can be imposed so that a secondary phase may be associated only if there is an associated primary phase from the same station (*primary_required_for_secondary*). Arrivals grouped by *StaPro* may not be associated if the group includes a regional S (specified by *regional_S_phases*) in addition to the primary arrival and a coefficient α times the (S–P) time is less than the distance (in degrees) from the station to the center of the grid cell (the α coefficient is currently set to 0.1). This restriction prevents association of regional primary phases at teleseismic distances.

A simple process based on travel time and slowness vector (if the latter is available) screens potential corroborating arrivals. The theoretical arrival time is estimated from the observed arrival time, the travel time for the phase stored in the grid file, and an estimate of the event origin time derived from the driver, assuming the location is at the center of the cell. The travel-time residual must be less than the sum of the time uncertainty **arrival**.*deltim*, multiplied by parameter *sigma_time*, and the maximum travel time across the grid cell. The maximum travel time across the cell is computed by *GAcons* and stored in the grid file. The slowness vector test is the same as that applied to the driver described in the previous paragraph.

A chi-square statistical test is applied if an arrival successfully passes the initial screening. The chi-square test examines the compatibility of the corroborating arrival and the driver arrival, with a hypothesized event within the current grid cell. The test uses all available features of the arrivals including travel time, slowness vector, and amplitude. A location is performed using the time and slowness vector of both arrivals, when available. Assuming that the data and model parameters are normally distributed, the location can be formulated as a least-squares problem, which is solved using the QR decomposition method, where matrix Q is orthogonal and matrix R is upper triangular. The matrix elements for the least-squares location problem are stored in the grid file for each grid cell. They consist of the spatial derivatives of time, slowness vector components, and magnitude. The model vector includes the origin time, magnitude (when amplitude data are available), and location (x, y, and z) of the hypothesized event. The data vector is made up of the two travel-time differences, the two magnitude residuals (when available), four components of the slowness vector (when available), and the x, y, and z components of the location. The location coordinates in the data vector help stabilize the inversion. The model and data vector components are scaled by their standard deviations, so that the sum of squares of residuals can be interpreted as a chi-square sum, which is compared to a user-specified threshold (*chi_limit*). The values for the standard deviations are important because they influence the chi-square sum. The values currently used for the standard deviations are the radius of the grid cell for the x and y components of location, the half-depth of the cell for the z component, the estimated standard deviation (**arrival**.*deltim*) for the two time measurements, 0.3 magnitude units for the amplitude measurement and the magnitude, and the estimated standard deviation for the slowness vector components (**arrival**.*delslo*).

The association process can be modified by limiting the corroborating associations to arrivals of the same technology using the Boolean parameters *hydro_corroborate_hydro_only* and *infra_corroborate_infra_only.* When set to a value of

TRUE, these parameters prevent mixed technology event hypotheses from forming at the onset of the association process. This is the current default setting at the PIDC and IDC.

Applying Weighted-count Event Confirmation Criteria

After all of the possible corroborating arrivals are examined for compatibility with the driver, a weighted-count confirmation test is applied to the preliminary event. The following features, multiplied by user-specified weights, are summed for the set of associated arrivals:

- time for primary phase (*primary_time_weight*)
- time for secondary phase (*secondary_time_weight*)
- time for hydroacoustic phase (*hydro_time_weight*)
- time for infrasonic phase (*infra_time_weight*)
- azimuth at array (*array_azimuth_weight*)
- azimuth at three-component (3-C) station (*3comp_azimuth_weight*)
- azimuth at hydroacoustic group (*hydro_azimuth_weight*)
- azimuth at infrasonic array (*infra_azimuth_weight*)
- slowness at array (*array_slow_weight*)
- slowness at 3-C station (3*comp_slow_weight*)
- slowness at hydroacoustic group (*hydro_slow_weight*)
- slowness at infrasonic array (*infra_slow_weight*)

The event hypothesis is removed from the list of preliminary event hypotheses if the sum is less than a user-specified threshold set by the parameter *weight_threshold*.

Applying Arrival-Quality Event Confirmation Criteria

The arrival-quality test is based on empirical observations and reduces the number of false alarms as a percentage of the total number of preliminary event hypotheses formed by GA. The test involves the individual detection quality as measured by the **arrival**.*delslo* parameter and the distance between the event hypothesis and the station. In the f-k plane, the **arrival**.*delslo* parameter can be thought of as the width of the incoming beam at the station. The larger the width, the wider the intersection of the beam with the surface of the earth at the event location. The intersection of the beam with the surface is proportional to the square of the **arrival**.*delslo* parameter ($\delta\theta$). The other parameter involved in the measure of the intersection of the beam with the earth's surface is the distance between the event hypothesis and the station (Δ). The intersection area is proportional to x = $\sin(\Delta/2)\ \delta\theta^2$. Figure 4 shows the geometry of the beam and surface intersection for straight rays.

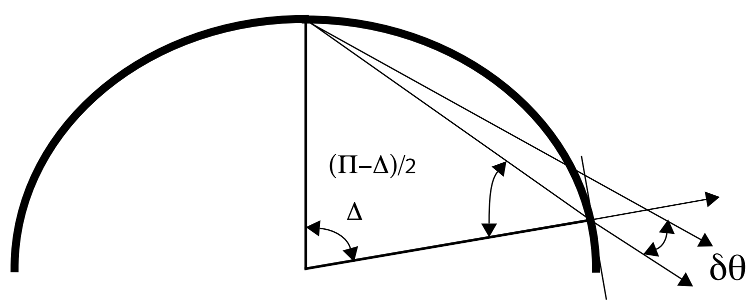


FIGURE 4. INTERSECTION OF STRAIGHT BEAM WITH SURFACE OF EARTH

The quantity x varies between 0 and a large positive number. A rational function is used to normalize x between a minimum of β (*ar_qual_beta*) and a maximum of 1. To account for the velocity variations, a distance dependence is built into the arrival quality expression:

$\Delta < 90^{o}$

$$Q(\delta\theta, \Delta) = \beta + \frac{1-\beta}{1+\alpha(\delta\theta^2 \cdot \sin(\Delta))}$$

$90^{o} < \Delta < 100^{o}$

$$Q(\delta\theta, \Delta) = \beta + \frac{1-\beta}{1+\alpha(\delta\theta^2)}$$

$142^{o} < \Delta < 148^{o}$

$$Q(\delta\theta, \Delta) = \beta + \frac{1-\beta}{1+\alpha\left(\frac{\delta\theta^2}{2}\right)}$$

elsewhere:

$$Q(\delta\theta, \Delta) = \beta + \frac{1-\beta}{1+\alpha\left(\delta\theta^2 \cdot \left(3 - \frac{(\Delta - 100)}{80}\right)\right)}$$

The arrival-quality test is applied to preliminary event hypotheses. The sum of the arrival qualities for the event hypothesis is computed and compared to a threshold value. The default values used for α (*ar_qual_alpha*), β (*ar_qual_beta*), and the threshold (*ar_qual_thresh*) are 0.75, 0.1, and 1.35, respectively; the value of the threshold is lowered to 1.2 in SEL1. These values were determined empirically to optimally eliminate false alarms while retaining analyst events. At this stage in the processing of preliminary event hypotheses, the distance used is the distance between the station and the center of the grid cell.

The arrival-quality test is applied to hydroacoustic phases as well, but has no screening effect other than allowing purely hydroacoustic event hypotheses with more than three phases to pass the test. Parameters α (*ar_qual_alpha_hydro*) and β (*ar_qual_beta_hydro*) are set by default to 0.0 and 0.46, respectively. For infrasonic phases, the values of these coefficients are 0.35 and 0.0008, respectively (*ar_qual_alpha_infra* and *ar_qual_beta_infra*). Applying the test to infrasonic phases prevents associations of phases at very large distances. The test supplements the maximum range limit specified in the file set by the parameter *dist_depth_range_file*. For infrasonic phases, the maximum range is set to 60 degrees.

Extracting Large Events

The presence of a large event (detected at many stations) in a data set typically generates numerous preliminary event hypotheses clustering in space and time around the location and origin time of the actual event. Large event hypotheses are extracted from the event hypothesis search space to reduce the processing load and eliminate numerous conflicts efficiently. A recursive search in terms of defining phases is used to find the largest event hypothesis. If the event hypothesis has more than the *min_ndef_extract* defining phases, a location is estimated. If the large event hypothesis survives location and outlier analysis with a sufficient number of defining arrivals (*min_ndef_extract*), the associated phases are removed from any other event hypotheses. This resolves potential conflict in favor of the extracted large event hypothesis. A further step of prediction of defining phases is performed after the large event is extracted.

Eliminating Redundant Event Hypotheses

Redundant event hypotheses are event hypotheses that are identical to other event hypotheses or a subset of other event hypotheses. Processing of redundant event hypotheses is done in several steps: splitting degenerate hypotheses, generating alternate regional group hypotheses, and eliminating redundant event hypotheses.

Splitting Degenerate Hypotheses

The initial event hypothesis construction can produce preliminary event hypotheses that are degenerate. Degenerate hypotheses contain incompatible arrivals, such as two or more arrivals at the same station identified as the same phase, or the same arrival identified as two or more different phases. When this occurs, the degeneracy is split into two or more separate, self-consistent event hypotheses. The weighted-count event confirmation test is reapplied after the split, because some of the resultant event hypotheses may no longer pass the test.

Generating Alternate Regional Group Hypotheses

Because the initial phase identification for regional phases may be incorrect, multiple interpretations of phases within a regional group are allowed. Several preliminary event hypotheses are generated when the regional group has two named phases.

The following phase name interpretations are allowed: Lg may be interpreted as Sn or Rg, Sn may be interpreted as Lg, and Rg may be interpreted as Lg.

Eliminating Redundant Event Hypotheses

The same set (or a subset) of associations can be consistent with two adjacent grid points. Consequently, a test is applied after splitting the hypotheses to remove the redundancies from the list of preliminary event hypotheses (*redundancy_required*). For an event hypothesis to be deemed redundant, the arrivals must form a subset of another event hypothesis's arrivals, the arrivals in common must be identified as the same phase, and the driver must be the same for both preliminary event hypotheses. In the case of identical sets of associations, the event hypothesis that best fits its grid cell is retained.

To test whether event hypothesis *A* is redundant with another, its associated arrivals are examined, and the arrival associated with the smallest number of event hypotheses is identified. If that number is one, event hypothesis *A* cannot be a subset of another. Otherwise, event hypotheses connected to the arrival are examined and their arrival sets are compared to that of event hypothesis *A*. If the arrival set of event hypothesis *A* is found to be a subset of one of these event hypotheses, then event hypothesis *A* is redundant.

As an efficiency step, a partial redundancy analysis is performed during the association of corroborating arrivals if the number of preliminary event hypotheses formed reaches a specified limit (*count_limit*). The redundancy analysis is only applied to preliminary event hypotheses built since the prior redundancy check.

Locating and Confirming Preliminary Event Hypotheses

Event hypotheses remaining after the preliminary screening tests described in the previous sections can be located and an analysis can be made of the residuals. Outliers are removed if necessary, redundancy checks performed, and the location refined. The locator uses the *libloc* library locator subroutine. The parameter *loc_fix_depth* controls whether or not the location is computed at fixed or free depth. The fixed depth option uses the depth of the center of the grid cell. The location residuals are analyzed in terms of a chi-square test. An outlier is defined as an arrival whose chi-square value is larger than a user-set threshold (*chi_outlier*). The following procedure is then applied to each preliminary event hypothesis to eliminate outliers:

1. Analyze outliers.

   If there are no outliers, proceed with event confirmation tests.

   If there are outliers; eliminate the worst outlier not including the driver. Eliminate the worst arrival tagged `wc_restricted` (arrival from auxiliary station) before eliminating any other defining arrival. Do not eliminate the arrivals tagged as `locked_association` under any circumstances.

   Re-locate the event hypothesis with the reduced set of arrivals.

   Apply the outlier analysis recursively until no outlier remains in the final event hypothesis.

2. Test redundancy.

   Apply the redundancy test after location and outlier analysis. This test differs from the test applied after the association of preliminary event hypotheses in that the driver is no longer required to be the same for both event hypotheses.

3. Apply event confirmation criteria.

   Apply event confirmation tests after location. These tests include the same weighted-count test that is applied after association of corroborating phases, a supplemental restriction that the event hypothesis should have a user-specified minimum number of associated arrivals (*req_num_of_defining_detections*), a restriction that the major axis of the error ellipse be less than a user-specified threshold (*max_smajax*), and an arrival-quality test based on uncertainty on the slowness measurement uncertainty (**arrival**.*delslo*) and the distance between the event hypothesis and the station (*ar_qual_thresh*).

4. Calculate probability-of-detection.

   Apply a network probability-of-detection test to preliminary event hypotheses after the location is computed. The test is intended to remove small event hypotheses that were not detected at a sufficient number of stations. Apply the test only to event hypotheses with less than a user-specified limit (*max_obs_net_prob*) of associated primary phases. The default value for *max_obs_net_prob* is 5.

   The network probability-of-detection test determines whether or not the set of stations at which primary phases have been detected is compatible with the location and magnitude of the preliminary event hypothesis. The probability-of-detection at each station is computed from the estimated location and magnitude of the event, the location of the station, nominal values of the station noise, snr detection threshold and reliability (*nois*, *noissd*, *snthrsh*, and *rely* are the corresponding attributes read from the **siteaux** database relation), and an amplitude attenuation table (the *atten_file* parameter is the directory for the attenuation table).

The test computes the difference between the log of the product of the likelihoods for all stations in the network and its expected value and normalizes this difference by the square root of the variance of the expected value. The hypothesis is eliminated if the computed value exceeds a user-specified threshold (*residual_over_sigma_max*). More detail on this test is available in [IDC5.2.1].

In applying the probability-of-detection test, only primary stations that have at least one detection in the time interval are considered.

The network probability-of-detection test that was applied before location is applied at this point (*probdet_after_location*), using the computed event location and magnitude.

Resolving Conflicts

An arrival may be associated to more than one event hypothesis up to this point in the processing sequence. The process that removes this ambiguity is called conflict resolution. Two methods are used to resolve these conflicts:

- Cluster analysis
- Resolution via association-based method

Cluster Analysis

The first method used in conflict resolution applies a clustering technique to identify and resolve conflicts between large event hypotheses that share a high percentage of associated arrivals. *GAassoc* frequently generates numerous small variations of a large event hypothesis. Experience has shown that association-based conflict resolution has difficulty distinguishing between these variations and tends to split a large event hypothesis into multiple smaller event hypotheses. Cluster analysis provides a means of identifying a group of similar preliminary event hypotheses and reducing that group to the single "best event." The "best event" is the one with the largest number of defining phases; the size of the error ellipse is used to break ties. The clustering method is only applied to clusters of large event hypotheses (that is event hypotheses with many defining phases) where it is likely that all members represent the same event hypothesis.

The following are key steps of the clustering algorithm:

1. Select the preliminary event hypothesis with the largest number of defining phases that has not already been clustered.

   The number of defining phases must be greater than or equal to a user-specified limit (*cluster_min_ndef*). If several event hypotheses have the same number of defining phases, then select the one with the smallest error ellipse. Call this event hypothesis the "best event."

2. Form a cluster by identifying all other preliminary event hypotheses that have at least a specified percentage (*cluster_min_pct_overlap*) of associations in common with the "best event."

   These associations must be time-defining, meaning that the arrival time is used in the calculation of the event location. The phase identifications may be different for the different preliminary event hypotheses.

3. Dissolve all preliminary event hypotheses in the cluster except for the "best event."

Continue as long as there are preliminary event hypotheses with a sufficient number of defining phases.

Conflict Resolution via Association-based Method

Conflicts that remain after cluster analysis are resolved by an association-based method. Each arrival associated with more than one event hypothesis is assigned to the event hypothesis that maximizes a weighted product of the goodness-of-fit and a measure of the quality of the event solution and other associations are removed. The goodness-of-fit is based on time, azimuth, slowness, and log-amplitude residuals. The event quality is based on the number of defining observations, size of the error ellipse, distance to the nearest station, probability of detection, and a factor introduced to help retain small event hypotheses. The test is applied iteratively; after each disassociation of an arrival all event hypotheses are relocated and all measures are recomputed.

The conflicting arrival is assigned to the event hypothesis that maximizes the quality measure:

$$L_{ij} = F_{ij}^{a} \cdot Q_i^{(1-a)} \tag{4}$$

where F_{ij} is a measure of the goodness-of-fit of the j^{th} arrival to the i^{th} event hypothesis, Q_i is a measure of the quality of the i^{th} event hypothesis, and a is a user-specified weighting factor (*master_tradeoff_weight*). If a is zero then the goodness-of-fit of the arrival to the event solution is ignored, and conflicts are resolved in favor of the event hypothesis with the highest event quality. Conversely, if a is set to one, then conflicts are resolved only on the basis of goodness-of-fit. Both F_{ij} and Q_i are normalized between 0 and 1, so L_{ij} also varies in this range. User parameters affecting the F_{ij} and Q_i are described in the sections that follow.

Determining Goodness-of-Fit

The goodness-of-fit of an association, F_{ij}, is based on the time, azimuth, slowness, and log-amplitude residuals when available. A χ^2 value is computed as:

$$\chi^2 = \sum_j ((d_j - m_j)/\sigma_j)^2 \tag{5}$$

where d_j is the observed data, m_j is the theoretical data, and σ_j is the estimated standard deviation of the j^{th} datum. The χ^2 is unbiased by using the equation:

$$\chi^2 = \chi^2 \cdot Ntot/(Ntot - Ndata) \tag{6}$$

where *Ntot* is the total number of data used to compute location and magnitude in the event hypothesis minus the number of model parameters in location and magnitude, and *Ndata* is the number of data for the arrival in question. The quantity F_{ij} is the probability corresponding to this unbiased χ^2 value.

Determining Event Quality

The quality of an event hypothesis, Q_i, is the normalized weighted sum of two terms:

$$Q_i = (b \cdot Q_{1i} + c \cdot Q_{2i}) / (b + c) \tag{7}$$

The first term Q_{1i} is a measure of the quality of the event solution based on the number of defining phases, the size of the error ellipse, the distance to the nearest station, and the probability of detection. The second term Q_{2i} increases the event quality measure if it is likely that the event hypothesis would be dissolved without the association (that is, if the number of defining phases is small). The factors *b* (*event_likelihood_weight*) and *c* (*dissolved_event_weight*) adjust the relative weight given to these two terms.

The first term, Q_1, is a measure of how likely the event hypothesis is to be real and is computed as the following normalized weighted sum:

$$Q_1 = \sum_{j=1}^{NA} w_j \cdot M_j \Big/ \sum_{j=1}^{NA} w_j \tag{8}$$

where *NA* is the number of event-quality attributes, w_j is the weight assigned to each attribute, and M_j is a measure between zero and one of how likely the event hypothesis is to be real based only on the j^{th} attribute. The M_j are approximated as linear ramps from 0.0 to 1.0 between interval bounds that are specified for each attribute.

The following event-quality attributes are used to compute Q_1:

- The number of defining phases (**origin**.*ndef*). If an event hypothesis has a large number of defining phases then it is more likely to be real than if it has a small number of defining phases. The interval bounds for M_j are specified by the user parameters *ndef_no_confidence_bound* (the default is 3) and *ndef_high_confidence_bound* (the default is 10). The weight is specified by the parameter *ndef_weight* (the default is 1.0).

- The size of the error ellipse (**origerr**.*smajax*). If the error ellipse is small then there is reasonable network coverage and the event hypothesis is more likely to be real than if the error ellipse is large. The interval bounds for M_j are specified by the parameters *smajax_no_confidence_bound* (the default is 500 km) and *smajax_high_confidence_bound* (the default is 10 km). The weight is specified by the parameter *smajax_weight* (the default is 0.8).
- The distance to the nearest station. If the nearest station is close, then the event hypothesis is more likely to be real than if it is far. The interval bounds for M_j are specified by the parameters: *dnear_no_confidence_bound* (the default is 90 km) and *dnear_high_confidence_bound* (the default is 10 km). The weight is specified by the parameter *dnear_weight* (the default is 0.5).
- Probability-of-detection. If the network probability-of-detection estimate is consistent with the set of stations that detected the event, then the event hypothesis is likely to be real. The interval bounds are applied to the ratio of the residual used in the probability of the detection event confirmation test and its standard deviation and are specified by the parameters *probdet_no_confidence_bound* (the default is 3.0) and *probdet_high_ confidence_bound* (the default is 1.0). The weight is specified by the parameter *probdet_weight* (the default is 0.7).

The second term in Q_i, Q_{2i} reduces the possibility of dissolving a small event hypothesis and is defined in terms of the number of defining phases as:

Q_{2i}=`0.0` if the number of defining arrivals is greater than or equal to *ndef_not_likely_to_dissolve_event* (the default is 6).

Q_{2i}=`1.0` if the number of defining arrivals is less than or equal to *ndef_which_will_dissolve_event* (the default is 3).

Q_{2i} is linear between these two bounds.

Iterative Procedure for Resolving Conflicts

This section describes the iterative procedure for applying the metrics to resolve conflicts. The Q_i are computed for all event hypotheses, and the L_{ij} are computed for each of the conflicting associations. The following tasks are performed once for each event hypothesis with conflicting associations:

1. Rank all conflicting associations based on their quality measure, L_{ij}. The rank is set to zero for the event hypothesis with the highest L_{ij}, and it is greater than zero for all other event hypotheses. Select the event hypothesis that has at least one conflicting association, whose rank is zero, and that has the highest proportion of defining associations that it is likely to keep after conflict resolution. This proportion is (*n1*+*n2*)/*ndef,* where *n1* is the number of defining associations that are not in conflict, *n2* is the number of conflicting associations with rank equal zero, and *ndef* is the total number of defining associations for the event hypothesis. To break a tie, select the event hypothesis with the highest event quality, Q_i.
2. For the current event hypothesis, select the conflicting association with the highest quality measure, L_{ij}, and disassociate this arrival from all other event hypotheses.
3. Relocate all event hypotheses that have lost an association, and reapply the event confirmation criteria. Abandon any event hypotheses that no longer satisfy these criteria, and disassociate all of their arrivals. Recompute Q_i and L_{ij} for all event hypotheses that have lost an association, and re-rank all affected associations. Return to Task 2 if the current event hypothesis has remaining conflicting associations with rank equal zero.

Several iterations through this procedure may be required to resolve all conflicts. The example below shows such a case.

Example

Figures 5 and 6 show a simple example of association-based conflict resolution. The examples start with four event hypotheses that have three conflicting arrival associations. Figure 5 shows the event hypotheses (A, B, C, and D), their associated arrivals, the conflicts (α, β, and δ), and the rank of the association (number). For simplicity in this example the association ranking is not modified after the event hypotheses are relocated.

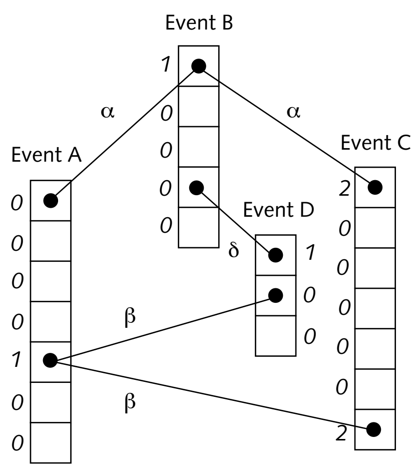


FIGURE 5. ASSOCIATION-BASED CONFLICT RESOLUTION

Two iterations of the procedure are required to resolve all conflicts in this example. Figure 6 shows the results at intermediate stages. The following procedure describes the first iteration; the tasks correspond to the procedure described in the previous section:

- Task 1: Select Event A.
- Task 2: Disassociate arrival α from Events B and C.
- Task 3: Relocate Events B and C. Both still pass event confirmation (Figure 6 [a]).
- Task 4: Select Event B.

- Task 5: Disassociate arrival δ from Event D.
- Task 6: Abandon Event D because it does not satisfy event confirmation. Re-rank conflicting associations of arrival β (Figure 6 [b]).

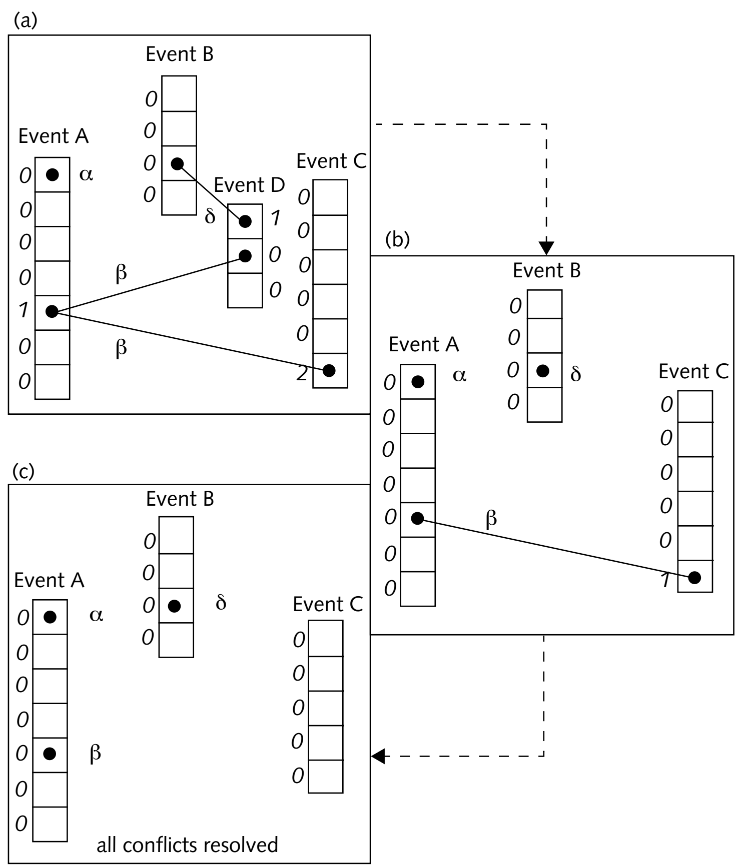


FIGURE 6. ASSOCIATION-BASED CONFLICT RESOLUTION PROCEDURE

Event C is not selected in the first iteration because it does not have any conflicting arrivals with rank equal to zero. Event D is not selected because it is dissolved during the processes of resolving the conflicts with Event B. The second iteration must resolve only one conflict:

- Task 1: Select Event A.

- Task 2: Disassociate arrival β from Event C.
- Task 3: Relocate Event C (Figure 6 [c]).

All conflicts are resolved after the two iterations, and Events A, B, and C remain. The re-ranking after each conflict is resolved as an important step in the procedure. In this example, arrival β would have been disassociated from Event A if the ranks were not updated.

Predicting Non-defining Phases

For each event hypothesis and each associated station, the travel time and predicted slowness and azimuth are obtained for each phase in the list of predicted phases specified by the parameter *predicted_phases*. A chi-square value is computed from the residuals computed as the difference between expected travel time, slowness, and azimuth and the measured values for a predicted arrival. If the chi-square value falls within the limits established by the parameter *chi_outlier*, the predicted phase is added to the association set for the event hypothesis.

Several detections may be candidates for a single phase. It is also possible for one detection to be identified as several phases. If either one or both of these situations occur, the detection-phase identification (ID) pair with the best fit based on its chi-square value is added to the association set.

Predicting Defining Phases

The differences between predicting defining phases and predicting non-defining phases are as follows:

- The error ellipse is taken into account for predicting defining phases, increasing the search time window for the predicted phases, but it is not used when predicting non-defining phases.
- Defining phases are predicted for stations that are not already associated; non-defining phases are predicted only for associated stations. The list of phases for which a prediction is attempted is set by the parameter *defining_phases*.

- The event is relocated and outlier analysis performed if a new defining phase is added. An incoming event should not be lost through the prediction of a defining phase. The initial state of the association is saved before location and outlier analysis, so that it can be recovered should the event be lost after adding the predicted defining phases.

Performing Seismological Checks

A number of physical consistency checks are performed before writing the conflict-free solutions to the database. When an event location is modified after failing one of the tests, the event hypothesis is resubmitted for all tests. This could lead to an infinite loop if no solution were found that satisfied all the tests. To avoid an infinite loop, the maximum number of iterations is limited. After an event hypothesis has been subjected to this maximum number of iterations, it is written to the database, even though it may not pass one of the required conditions.

Perform the following checks:

- Validate the coda phase name.

  The software checks that non-defining regional phases are identified as Sx and that all non-defining coda phases are properly identified as tx, Px, or Sx.

- Ensure that there is a single hydroacoustic phase at hydroacoustic stations.

  The software eliminates cases where both T and H phases are associated at the same hydroacoustic station.

- Check deep event hypotheses.

  The software checks deep event hypotheses to make sure that none occur in areas where no deep seismicity is known to occur. One of the grid files generated by *GAcons* contains the cell geometry used in this check. The grid file is specified using the parameter *gridfile*. When a deep event hypothesis occurs outside of any deep seismic cell and its confidence ellipse does not intersect any deep seismic cell, the software attempts to relocate the event hypothesis at the surface. If this fails, it removes the event hypothesis.

- Check distance-depth ranges.

  Seismic and acoustic phases are constrained in distance and depth, as specified in a file specified by the parameter *dist_depth_range_file*. The software checks all associations against these ranges. If an arrival is found to be out of its range, its phase name is changed to one that falls in the correct range. The list of potential alternative phase names is given in the parameter *equivalence_list*.

- Complete hydroacoustic groups.

  The software checks that all hydroacoustic arrivals belonging to a hydroacoustic group are associated.

- Screen out non-defining predicted phases if their residuals are large.

  If large residuals for a combined time and slowness measure are found for non-defining phases, the software removes the corresponding association from the association set. The criterion used to judge the size of the residual is a chi-squared test, and the threshold is set by the parameter *chi_outlier*.

- Check regional pairs.

  Phase combinations Pn-Rg and Pg-Sn are not allowed if no other regional phases are included at the station. If such a combination is encountered at a station, the software changes the name of the secondary phase, and the pairs become Pn-Lg or Pg-Lg.

- Remove isolated secondary phases.

  If the Boolean parameter *primary_required_for_secondary* is set to TRUE and an isolated secondary phase is present at a station without a primary phase, the software removes the association corresponding to the isolated secondary phase.

- Remove duplicate phases.

  If duplicate phases exist at the same station, the software keeps the phase with the smallest residual and removes the other from the association list.

- Check for incompatible M_L and m_b.

  To avoid spurious M_L magnitudes that are very different from m_b for the same event hypothesis, the software checks event hypotheses with a well-determined m_b magnitude. The two criteria for determining if m_b is well determined are a minimum number of contributing stations, with a threshold set by the parameter *min_sta_magnitude_diff*, and the maximum uncertainty on the m_b calculation, which is set by the parameter *max_uncertainty_ magnitude_diff*. If the event m_b is well constrained, the software disassociates all arrivals at a station whose M_L is such that the absolute difference M_L–m_b is larger than *default_max_magnitude_diff*. The default value specified by this parameter may be overridden by a set of station-specific values specified by the parameter *max_magnitude_diff*.

- Check small, deep event hypotheses.

  At this stage of the processing, the software relocates deep event hypotheses with fewer than *max_ndef_reset_fix_depth* defining phases to the surface of the earth using a constrained depth algorithm. If a consistent surface location can be obtained, the preliminary event hypothesis is kept at the surface; otherwise, the original deep location is kept.

- Check the order of the phases.

  The software imposes some phase-ordering constraints using the parameter *phase_ordering_list*. If two phases are in reverse order, they are switched if they are both secondary phases, or the secondary phase is eliminated if the other is a primary.

- Check for magnitude outliers.

  If the uncertainty of the calculated station m_b magnitude is larger than the value set by *max_reported_mag_sig*, the software makes the magnitude non-defining at that station.

MAINTENANCE

The maintenance of the GA software consists of purging log files, keeping grid files current, and adding or deleting stations, as needed.

Purging Log Files

The log files created by GA in normal pipeline operations are purged at the same time as log files from other applications are purged. At the IDC, log files are kept online for a period of 10 days.

Updating GA grid Files

Update the GA grid files when you modify the primary S/H/I networks or any of the propagation characteristics, such as the travel-time tables. This ensures optimal processing at the event formation stage, where the stations closer to the current grid cell are systematically searched for driver arrivals.

Adding New Stations

The addition of new stations to one of the networks should trigger the start of the *GAcons* utility to ensure that the PKB grid file is complete, with the inclusion of the propagation information for that station. To ensure that the *GAcons* run is effective, update the appropriate tables, including the rows for the **site**, **siteaux**, and **affiliation** tables and run *GAcons* with the appropriate jdate.

Seismic Station

To add a primary seismic station to GA processing:

1. Insert a row for that station into the **site**, **siteaux**, and **affiliation** tables.
2. Run the *GAcons* utility to create a grid file including the new station. You do not need to update the GA parameters other than the *jdate* parameter which should be set to the current date. The station is automatically included in the grid file if it is part of the network specified by the *GAcons* parameter *net*.

3. After *GAcons* has been run, check the grid file by using the GUI utility *GAgrid* to make sure that the station was included in the file.

Hydroacoustic Station

To add a hydroacoustic station to GA processing:

1. Insert a row for that station into the **site**, **siteaux**, and **affiliation** tables. If the station is part of a group from a multi-site station, two rows of the **affiliation** table need to be added: one for the hydroacoustic stations network and one for the group to which the station belongs.

   When adding a hydroacoustic station, the blockage file for that station needs to be added in the directory indicated by the parameter *blockage_spec_dir*.

2. If seasonally-varying 2-D velocity maps are available for that station, place them in the directory indicated for that purpose in the file pointed to by the velocity specification parameter *vmodel_spec_file*. The second line of that file indicates the directory for these velocity maps.

Infrasonic Station

To add an infrasonic station to GA processing:

1. Insert a row for that station into the **site**, **siteaux**, and **affiliation** tables.
2. If seasonally varying 2-D velocity maps are available for that station, place them in the directory indicated for that purpose in the file pointed to by the velocity specification parameter *vmodel_spec_file*. The second line of that file indicates the directory for these velocity maps.

Chapter 3: Troubleshooting

This chapter describes how to identify and correct problems related to GA and includes the following topics:

- Monitoring
- Interpreting Error Messages
- Solving Common Problems
- Reporting Problems

Chapter 3: Troubleshooting

MONITORING

GA requires the following main tools to verify the proper function of the software in normal operations: the *WorkFlow* monitoring tool, database queries, and the log files produced by all components of GA. These tools are described in the following paragraphs.

Processing Status Display

The *WorkFlow* monitoring tool allows the processing engineers to visualize the progress of Station Processing, Network Processing, and Post-analysis Processing. Figure 7 shows an example of the use of *WorkFlow* to monitor the three Network Processing pipelines at the IDC. The SEL3 pipeline in that figure shows the state of the most recent interval in SEL3 as `GAassoc-started`, indicated by the orange-colored brick, while the other bricks are in the `done` state, indicated by the dark green color. The `done` state signifies that the Network Processing pipeline has completed processing. The states relating GA processing status for which graphical feedback is available are `GAassoc-done`, `GAassoc-failed`, `GAassoc-retry`, `GAassoc-started`, `GAconf-done`, `GAconf-failed`, `GAconf-retry`, and `GAconf-started`, `GA_DBI-done`, `GA_DBI-failed`, `GA_DBI-retry`, and `GA_DBI-started`. The `GAassoc-started` state is the only one of these states shown in Figure 7. The `GAassoc-failed`, `GAconf-failed`, and `GA_DBI-failed` indicate a problem and are shown in red. The operator should report any occurrences of these failed states and attempt to manually reprocess the failed intervals.

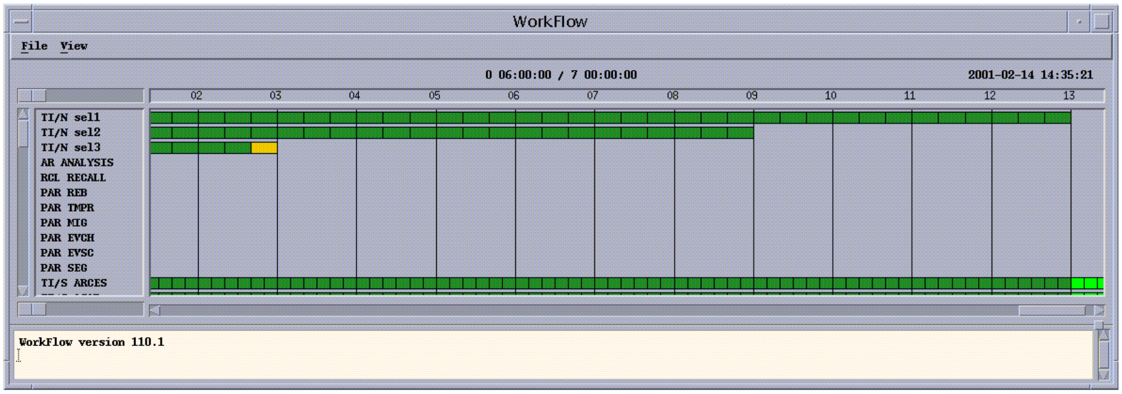


FIGURE 7. WORKFLOW DISPLAY SHOWING STATUS OF NETWORK PROCESSING PIPELINES

Querying the Database

GA produces an automatic event bulletin. One way to monitor the bulletin's progress and assess the quality of the bulletin produced is to monitor the writing and content of the tables that form the bulletin. These tables are the SEL*n*.**origin**, SEL*n*.**origerr**, SEL*n*.**assoc**, SEL*n*.**event**, SEL*n*.**netmag**, and SEL*n*.**stamag** tables, where *n* takes values from 1 to 3. The following is an example origin query:

```
SQL> select orid, &etoh, lat, lon, depth from sel1.origin
     where time > 984650400 ;
```

The epoch time in this example is the minimum time on the latest interval processed through SEL1. The results of the query output are as follows:

```
     ORID                ETOH        LAT        LON NDEF
--------- ------------------- --------- --------- ----
   572621 03/15/2001 10:00:40   71.8042   90.6632    3
   572618 03/15/2001 10:07:20  -14.1093  167.4894   14
   572617 03/15/2001 10:09:21  -26.3900   71.4506    5
```

The same query repeated 20 minutes later produces an expanded list of origins, where the origins formed in the latest 20 minutes are added to the output of the query.

```
SQL> SELECT orid, &etoh, lat, lon, ndef from sel1.origin
     where time > 984650400;

     ORID                ETOH       LAT       LON NDEF
--------- ------------------- --------- --------- ----
   572621 03/15/2001 10:00:40   71.8042   90.6632    3
   572618 03/15/2001 10:07:20  -14.1093  167.4894   14
   572617 03/15/2001 10:09:21  -26.3900   71.4506    5
   572642 03/15/2001 10:28:54   66.6424   22.0695    3
   572640 03/15/2001 10:35:50   47.7434   16.1521    3
   572638 03/15/2001 10:47:45   21.8068   37.9084    3
   572641 03/15/2001 10:56:43   61.5182 -154.2037    3
```

Log Files

Messages written by the *GA_DBI*, *GAassoc*, and *GAconflict* programs are directed to log files placed in the directory `$(LOGDIR)/`*jdate*`/GA`, where `LOGDIR` is defined for the site (for example, in `shared.par`) and *jdate* is the Julian date of the data day being logged. The logs record the progress of the processing and the potential errors that may occur during the processing. Monitor the logs for potential errors in network processing. The log files are most useful for obtaining additional information about a problem identified in *WorkFlow*.

INTERPRETING ERROR MESSAGES

GA is a complex subsystem, and a large number of informative and error messages can be produced by the system. The number and type of messages that are written can be controlled by a set of verbosity parameters. Table 4 shows the number of informative and error messages included in each of the software items in GA at different levels of reporting verbosity. The level of verbosity of the reporting is set by the user parameters *global_verbose*, *assoc_verbose*, *ev_verbose*, *cr_verbose*, *loc_verbose*, and *debug_verbose*. Each of these parameters can take a value from 0 to 4 corresponding to levels zero, low, medium, high, and very high. The numbers in the table are the total numbers of calls at particular levels of reporting. The default values for parameters are set to 1, except for *loc_verbose*, which has a value of 0. A high or very high verbosity level produces large log files and should only be used for debugging purposes. The following paragraphs describe a few of the error and reporting messages.

TABLE 4: NUMBER OF CALLS TO REPORTING SYSTEM

	Verbose Level				
Software Item	**Zero**	**Low**	**Medium**	**High**	**Very High**
GAcons	27	27	33	33	33
GAassoc	81	103	147	178	187
GAconflict	56	99	121	129	133
GA_DBI	13	16	26	26	26
libGA	11	54	117	122	125

Exit Errors

A number of conditions or fatal errors may cause the system to call the `exit()` function. The error codes returned by the `exit()` function are set through the use of the GA parameters shown in Table 5. When the DACS detects an abnormal exit condition, which is the case for all exit values listed in the table except for the *exit-ga-no-arrivals*, several (specified by the *tuxshell* parameter *failcount*) attempts are made to reprocess the interval in an attempt to recover from a transient problem such as a temporarily unavailable database. The log files reflect the reprocessing attempts and, unless the failure condition is due to a temporary condition, the same failure is logged each time. The error messages displayed on exit conditions are written for the lowest verbosity level.

TABLE 5: EXIT VALUES FOR GA_DBI, GAASSOC, AND GACONFLICT

Type of Error	Parameter Name for Exit Value	Current Value of Parameter
no arrivals to process	*exit-ga-no-arrivals*	`10`
error initializing or opening a database connection	*exit-gdi-open-failure*	`11`
error submitting a query to the database	*exit-gdi-submit-failure*	`12`
error allocating dynamic memory	*exit-ga-memory-allocation-failure*	`13`
error calling internal GA function	*exit-ga-error*	`14`

exit-ga-no-arrivals

The *exit-ga-no-arrivals* exit type occurs when there are no arrivals to process within the interval. The DACS recognizes this as a normal exit and does not flag this as an error. In *GAassoc*, two possible conditions cause this exit.

Message: `Start time greater than or equal to end_time`

Description: The start time of the interval is larger than the end time.

Action: None. This is a normal exit.

Message: `No arrivals to process. Exiting ...`

Description: There are no arrivals to process within the interval.

Action: None. This is a normal exit.

In *GAconflict*, no informative message is printed when there are no arrivals to process.

exit-gdi-open-failure

When a call to one of the GDI initialization functions (`gdi_init()`, `gdi_open()`, or `gdi_error_init()`) fails, a GDI error message is displayed, and the system exits with the value set by the parameter *exit-gdi-open-failure*.

exit-gdi-submit-failure

Message:	`while querying db using query=` *query text*
Description:	A database query failed. GA exits with the value set by the parameter *exit-gdi-submit-failure*.
Action:	Troubleshoot the problem using the query. In most cases, the faulty query makes the source of the problem clear. For instance, the name of a table within the query may be set incorrectly, misspelled, or missing.

exit-ga-memory-allocation-failure

Message:	`Error in allocating space for origin`
	`Error in allocating space for origerr`
	`Error in allocating space for assoc`

Description:	An error was encountered in a call from *GAassoc* to a memory allocation system subroutine. The system exits with the value set by the parameter *exit-ga-memory-allocation-failure*.
Action:	Depending on the exact cause of the failure, you may either be able to diagnose the cause of the problem, for instance if the processing interval is too long, or report the problem to the maintainer of the software.

Message:	`Error in allocating space for **all_phases`
	`Error in allocating space for orid_conf`
	`Error in allocating space for orid_num_prev`
	`Error in allocating space for orid_prev`
	`Error in allocating space for origins`
	`Error in allocating space for origerr`
	`Error in allocating space for assoc`

Description:	An error was encountered in a call from *GAconflict* to a memory allocation system subroutine. The system exits with the value set by the parameter *exit-ga-memory-allocation-failure*.
Action:	Depending on the exact cause of the failure, you may either be able to diagnose the cause of the problem, for instance if the processing interval is too long, or report the problem to the maintainer of the software.

exit-ga-error

The *exit-ga-error* type of exit condition occurs after an error other than database initialization, database access, or memory allocation occurs within one of the programs. In most cases, a message is displayed before the program exits. For each program, the following messages are displayed before this type of exit occurs:

Message:	`Problems encountered while trying to read from par file !!!`
Description:	An error was encountered in function `GA_read_par()`, which is called within *GA_DBI*, *GAassoc*, and *GAconflict*.
Action:	Check the parameter files involved and the command line setting the call to the programs.

The following messages may be found in a *GAassoc* log file:

Message:	`Problems while trying to read SASC tables. Exiting!!`
Description:	An error occurred while reading the SASC tables. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for the directory of the SASC files and the format of the SASC files.
Message:	`Error while trying to read atten file.  Exiting!!`
Description:	An error occurred while reading the attenuation file, as specified by the parameter *atten_file*. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for *atten_file*.
Message:	`No grid file specified.  Exiting ...`
Description:	An error occurred while reading the GA grid file. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for the grid file directory and name (*input_path* and *input_file*).
Message:	`Problems reading travel-time tables.  Exiting ...`
Description:	An error was returned from the `setup_tt_facilities()` function, which sets up the travel-time facilities. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for the velocity model (*vmodel_spec_file*).

Message:	`Problems reading blockage tables.  Exiting ...`
Description:	An error was returned from the `setup_blk_facilities()` function, which sets up the hydroacoustic blockage facility. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for the blockage files directory (*blockage_spec_dir*).

Message:	`Problem initializing magnitude system.  Exiting!!`
Description:	An error was returned from the `setup_mag_facilities()` function, which sets up the magnitude handling system. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for the directory of the magnitude tables (*magnitude_spec_file*).

Message:	`Error in GAarrivals`
Description:	An error was returned from the `GAarrivals()` function, which reads the arrival data. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Contact the maintainer of the software.

Message:	`FATAL ERROR !!! Non-existent value in the siteaux table in the database`
	`for at least one of nois, noissd, rely, snthrsh`
	`for at least one of the stations in the network`

Description:	An error was returned from the `GA_station_info()` function, which reads station information from the database. *GAassoc* exits with a value of *exit-ga-error*.
Action:	Check that the **siteaux** database table contains records for stations in the network.

Message: `FATAL ERROR in GA_active_stations`

Description: An error was returned from the `GA_active_stations()` function, which establishes the list of active stations. *GAassoc* exits with a value of *exit-ga-error*.

Action: Contact the maintainer of the software.

Message: `Fatal error, -1, in GA_prelocation_cluster()! Exiting!!!`

Description: An error was returned from the `GA_prelocation_cluster()` function, which performs cluster analysis before location. *GAassoc* exits with a value of *exit-ga-error*.

Action: Contact the maintainer of the software.

Message: `Fatal error, -1, in GA_cluster()! Exiting!!!`

Description: An error was returned from the `GA_cluster()` function, which performs cluster analysis after location. *GAassoc* exits with a value of *exit-ga-error*.

Action: Contact the maintainer of the software.

Message: `Fatal error, -1, in GA_assoc_based_CR()! Exiting!!!`

Description: An error was returned from the `GA_assoc_based_CR()` function, which performs association-based conflict resolution. *GAassoc* exits with a value of *exit-ga-error*.

Action: Contact the maintainer of the software.

The following messages may be found in a *GAconflict* log file:

Message:	`Problems while trying to read SASC tables. Exiting!!`
Description:	An error occurred while reading the SASC tables. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for the directory of the SASC files and the format of the SASC files.
Message:	`Error while trying to read atten file. Exiting!!`
Description:	An error occurred while reading the attenuation file set by the parameter *atten_file. GAconflict* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for *atten_file*.
Message:	`Problems reading travel-time tables.  Exiting ...`
Description:	An error was returned from the `setup_tt_facilities()` function, which sets up the travel-time facilities. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Check the parameter setting for the velocity model (*vmodel_spec_file*).
Message:	`Number of origerr tuples (12) different from number of origin tuples (13) in GA tables`
Description:	The number of **origerr** rows (tuples) is different from the number of **origin** tuples in the temporary GA tables. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Contact the maintainer of the software.

Message:	`Number of origerr tuples (12) different from number of origin tuples (13) in Final tables`
Description:	The number of **origerr** tuples is different from the number of **origin** tuples in the final tables. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Contact the maintainer of the software.

Message:	`Number of origerr tuples (12) different from number of origin tuples (13) in Previous tables`
Description:	The number of **origerr** tuples is different from the number of **origin** tuples in the previous tables. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Contact the maintainer of the software.

Message:	`Error in GAarrivals`
Description:	An error was returned from the `GAarrivals()` function, which reads the arrival data. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Contact the maintainer of the software.

Message:	`FATAL ERROR !!! Non-existent value in the siteaux table in the database`
	`for at least one of nois, noissd, rely, snthrsh`
	`for at least one of the stations in the network`

Description:	An error was returned from the `GA_station_info()` function, which reads station information from the database. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Check that the **siteaux** database table contains records for stations in the network.

Message:	`FATAL ERROR in GA_active_stations`
Description:	An error was returned from the `GA_active_stations()` function, which establishes the list of active stations. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Contact the maintainer of the software.
Message:	`ERROR in GA_build_drivers. Arrivals in the time interval provided may be incompatible with the origin and assoc tables in the GAassoc tables`
Description:	An error was returned from the `GA_build_drivers()` function when building the driver structures for the event hypotheses in the GA temporary tables. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Check that the start-hook to *GAassoc* that clears the temporary tables is properly set up, modify the setup if needed, and run the failed interval again.
Message:	`ERROR in GA_build_drivers. Arrivals in the time interval provided may be incompatible with the origin and assoc tables in the previous time interval from the Previous tables`
Description:	An error was returned from the `GA_build_drivers()` function when building the driver structures for the event hypotheses in the previous tables. *GAconflict* exits with a value of *exit-ga-error*.
Action:	Contact the maintainer of the software.

Message:	`ERROR in GA_build_drivers. Arrivals in the time interval provided may be incompatible with the origin and assoc tables in the previous time interval from the Final tables`
Description:	An error was returned from the `GA_build_drivers()` function when building the driver structures for the event hypotheses in the final tables.
Action:	Contact the maintainer of the software.

Non-exit Errors

Error conditions that do not cause the GA applications to exit are reported through the log files. These conditions are not fatal, and an informative message is printed when they occur. A few examples of these messages, written at the default verbosity levels and taken from the GA log files, are provided below:

Message:	`WARNING ........`
	`Removing event 562895`
	`ERROR relocating after modification due to a phase being out of range.`

Description:	A small event (three defining phases) was removed from the list of event hypotheses after an attempt at relocating with a phase named Pg instead of Pn failed. Pn was renamed Pg, because the event to station distance was not within the defined distance range for Pn.
Action:	Informative message, no action is necessary.

Message:	`WARNING ........`
	`Removing event 563838`
	`Failed shear phase test for close-in station.`

Description:	A two-phase event failed to pass the test that a station at a distance of less than 10 degrees should include a shear phase. The event was removed because it failed the event confirmation threshold test.
Action:	Informative message, no action is necessary.

SOLVING COMMON PROBLEMS

This section provides instructions for solving some common problems arising in the use of this software. These problems usually occur during installation and initialization of the software.

The most common error encountered during installation or while running GA offline is failure to establish a start-hook for *GAassoc* to trigger the script that clears the temporary database tables used as output by *GAassoc*. When the start-hook is not properly configured, the temporary tables are not purged before running *GAassoc*. This causes a problem with *GAconflict* where there are incompatibilities between the arrivals read into the process and the association sets read from the temporary tables. Error messages are written to the log file to inform the user about the incompatibilities. The error originates from a failure to clean up the temporary tables.

Another common error encountered in normal operations is the failure to add a new station to the GA grid file. You will observe a low association rate for arrivals from stations that are not included in the grid. Verify this problem by examining the grid using *GAgrid* and correct the problem by rebuilding the grid.

In stand-alone operations, common errors are:

- failure to set up all of the necessary database tables, especially the temporary tables and the **ga_tag** table

- failure to set up the configuration to point to the correct database tables and configuration files
- failure to point to the intended time period

These errors generally cause GA to misfunction and can generally be debugged by examining the log files.

Using the wrong, but valid, configuration files is a much more subtle problem that can only be avoided by careful configuration.

Error Recovery

If GA processing errors occur, examine the log files for the applications *GA_DBI*, *GAassoc*, and *GAconflict*, as well as the corresponding *tuxshell.* These log files are the primary source of information and diagnosis. The log files for the GA applications are in the directory `$(LOGDIR)/`*jdate*`/GA`, where the *jdate* is the Julian date of the data day where a failure occurred. The log files for the corresponding *tuxshell* are in `$(LOGDIR)/`*jdate*`/tuxshell`.

If you cannot diagnose and correct the cause of the failure, reprocess the interval(s) that failed. Then, if you cannot correct the failure, report it. If the error is due to external factors, such as a file server or database server, GA can be terminated and restarted from the point of failure. For optimal processing, any results written to the database after the failure should be purged and the intervals reprocessed in sequence rather than trying to fill the gap.

REPORTING PROBLEMS

The following procedures are recommended for reporting problems with the application software:

1. Diagnose the problem as far as possible.
2. Record information regarding symptoms and conditions at the time of the software failure.
3. Retain copies of relevant sections of application log files.

4. Contact the provider or maintainer of the software for problem resolution if local changes of the environment or configuration are not sufficient.

Chapter 4: Installation Procedures

This chapter provides instructions for installing the software and includes the following topics:

- Preparation
- Executable Files
- Configuration
- Database
- Initiating Operations
- Validating Installation

Chapter 4: Installation Procedures

PREPARATION

GA is delivered as a component of the overall IDC system. Prior to installation of the IDC software, make sure that there is sufficient disk space at the site of the installation to receive the delivery, that the hardware servers have been procured in sufficient quantity, and that all public domain and COTS software recommended to run the system have been installed on the Local Area Network (LAN) prepared to receive it. In the case of GA, it is particularly important to ensure that the correct version of the Solaris operating system and the ORACLE RDBMS are installed.

Obtaining Released Software

The software is obtained via File Transfer Protocol (FTP) from a remote site or via a physical medium, such as tape or Compact Disk–Read Only Memory (CD-ROM). The software and the associated configuration data files are stored as one or more tape archive (tar) files. The software and data files are first transferred via FTP or copied from the physical medium to an appropriate location on a local hard disk. The tar files are then untarred into a standard UNIX directory structure.

Hardware Mapping

The user must select the hardware on which to run the software components. Software components are generally mapped to hardware to be roughly consistent with the software configuration model.

EXECUTABLE FILES

Install the executable files by using the Makefiles provided with the software delivery. The following commands install the software:

```
% cd <dir>/src/automatic
% make install
```

The above commands with *<dir>* set to the appropriate directory install the executable binary files at the proper location within the operational tree. The *tuxshell* parameter files are set to point to this location using a UNIX variable so that the applications are called during normal operational processing.

CONFIGURATION

Parameter Files

The parameter files for the GA applications (with the exception of *GAgrid*) are located under *<dir>*`/app_config/automatic`, where the *<dir>* directory is site dependent (`/cmss/config` at the IDC) and set by a UNIX variable. The parameters are organized into five directories.

The first directory, *<dir>*`/app_config/automatic/GA`, contains the parameter file `GA.par` and the pipeline-specific parameter files `GAsel1.par`, `GAsel2.par`, and `GAsel3.par`. Most GA parameters are common to all processing pipelines as well as to the three programs, and their values are set in `GA.par`. The pipeline-specific parameter files include the `GA.par` parameters and contain the settings of table names as well as the parameters that are specific to each pipeline. For example, the *phases* parameter is specific to each pipeline; the *phases* parameter in the `GAsel2.par` file includes the infrasonic phases whereas the same parameter does not include the infrasonic phases in parameter file `GAsel1.par`.

The second directory, *<dir>*`/app_config/automatic/GA_DBI`, contains the parameter files `GA_DBI_sel1.par`, `GA_DBI_sel2.par`, and `GA_DBI_sel3.par`. These files are specific to each pipeline and are used mostly to set the table name parameters as well as parameters specific to the application *GA_DBI* (for example, parameters *hydro_snr_thresh* and *hydro_max_H_per_hour*). These three files include the `GAsel`*n*`.par` parameter file corresponding to their specific pipeline (*n*=`1`, `2`, or `3`).

The third directory, <*dir*>`/app_config/automatic/GAassoc` contains the parameter files `GAassoc_sel1.par`, `GAassoc_sel2.par`, and `GAassoc_sel3.par`. These files are specific to each pipeline and are used mostly to set the table name parameters.

The fourth directory, <*dir*>`/app_config/automatic/GAconflict` contains the parameter files `GAconflict_sel1.par`, `GAconflict_sel2.par`, and `GAconflict_sel3.par`. These files are specific to each pipeline and are used mostly to set the table name parameters.

The fifth directory, <*dir*>`/app_config/automatic/GAcons` contains one parameter file, `GAcons.par`, used by application *GAcons* to generate the GA grid files.

GA.par, GAsel1.par, GAsel2.par, GAsel3.par

Directory: <*dir*>`/app_config/automatic/GA`

The `GA.par` file contains parameters common to all pipeline GA applications (*GA_DBI*, *GAassoc*, and *GAconflict*) in all three pipelines. The `GAsel`*n*`.par` parameter files contain parameters valid for the SEL*n* pipeline.

The following example shows the `GA.par` parameter file:

```
#############################################################
# This file contains all GA parameters common
# across the pipelines and across the applications
# (GAassoc, GAconflict, GA_DBI)
#
#############################################################
#
# Exit codes for ga.
#
#
```

```
exit-ga-no-arrivals=10
exit-gdi-open-failure=11
exit-gdi-submit-failure=12
exit-ga-memory-allocation-failure=13
exit-ga-error=14
#
#
# Define environment variables for database tables
# to be shared by GAassoc, GAconflict
#
# GDI database parameters
#
vendor="oracle"
#
# Verbose parameters
#
global_verbose=1
cr_verbose=1
loc_verbose=0
ev_verbose=1
#
# GA grid file name specification
#
input_file="AA_IDC.sector.-180deg.to.180deg"
#
# Grid file name. This is the file containing geographic information
# about the GA grid. It is an abbreviated version of the large
# file used in GAassoc.
#
gridfile="AA_IDC.grid.3.all_sectors"
#
#
# Time stepping parameters
#
origin_time_limit=1200.
#
# Lookback setup for the hydro arrivals
#
end_time_hydroacoustic=11400.
end_time_infrasonic=61800.
```

```
end_time_seismic=3600.
#
# Set parameter to not corroborate hydro drivers
# with hydro arrivals
#
hydro_corroborate_hydro_only
infra_corroborate_infra_only
#
# Amplitude attenuation file
#
atten_file=$(ATTEN_DIR)/slowamp.P
#
# rely parameters to override siteaux table
#
ss_rely=0.8
ar_rely=0.9
#
input_path=$(GA_SITE_PATH)/
dist_depth_range_file=$(GA_SITE_PATH)/GA_dist_depth_ranges
#
# setting of configuration for Beta and Delta pipeline
#
noprev_efficiency
#
# Velocity model specification
#
vmodel_spec_file=$(VMSF)
magnitude_spec_file=$(MDF)
transmission_loss_spec_file=$(TLSF)
ml_magtype=ml
mb_magtype=mb
mag_atten_dir_prefix=$(ATTEN_PREFIX)
blockage_spec_dir=$(STATICDIR)/BLK_OSO
#
# modeling error deltim
#
dist_var_wgt
#
#
# Station-dependent slowness and azimuth corrections
```

```
#
# NOTE - set in IMSPAR, so not re-set here
# sasc_dir_prefix=
#
#
# Character strings used in the site table entry statype
# to specify arrays.
#
array_types="ar,hfa,spa"
#
# Character string(s) that specify hydro stations
# for possible future use in the site table.
#
hydro_types="hy"
infra_types="is"
#
# Amplitudes parameters
#
# arrival_amptype          [default="A5/2"]
ml_amptype=SBSNR
#
# Restriction on distance range for mb magnitude
#
max_magdist_body=98.
min_magdist_body=20.
# max_depth_ml=40.         [default value=40.]
#
# Default channel name of sbsnr table
#
default_sbsnr_chan="V2040"
#
# Suffix for file name containing mb corrections
#
mb_dist_depth_suffix="qfvc"
maxrecs=200000
#
# Channel name from siteaux table
#
channels="cb,bz,sz,ed,ez,is"
#
```

```
defining_phases="P,PKP,Pn,Pg,Lg,Sn,Rg,H,I"
primary_phases="P,PKP,Pn,Pg"
regional_phases="Pn,Pg,Sn,Lg,Rg"
regional_S_phases="Sn,Lg,Rg,Sx"
hydro_phases="H,T,O,Hx,Tx,Ox"
infra_phases="I,Ix"
driver_phases="P,Pg,Pn,I,H,O"
alias_list="(O H)"
hydro-noise-phase=N
noise-phase=N
unknow-phase=-
#
# Forward transformation list. The first phase in each group
# specifies the initial phase name. The following phase names
# indicate what phases this phase can be transformed into
#
forward_transformation_list="(P PKP Pdiff Pn S ScP PcP PKPab\
  PKPbc PP ScS),(S Rg Sn Lg),(Pn P Pg Pdiff S ScS),(Lg Sn Rg),\
  (Sx Sn Lg Rg S ScS),(tx PcP PKPbc PKPab),(Rg Lg),(Sn Lg S),\
  (Pg Pn),(H T)"
#
primary_required_for_secondary
#
# Optional processes parameters
#
redundancy_required
do_clustering
do_association_based_conflict_resolution
#
# Association loop parameters.
#
num_first_sta=5
count_limit=200000
chi_limit=.99
chi2_dist_coef=6.
#
# Prelocation Cluster analysis parameters
#
nobefore_location_cluster
preloc_cluster_min_ndef=8
```

```
preloc_cluster_min_pct_overlap=0.8
#
# Screening parameters adjustment
#
sigma_time=3.
sigma_slowness=3.
belief_threshold=1.
#
#location parameters
#
loc_conf_level=0.90
noloc_fix_depth
chi_outlier=.99
#
# Event confirmation criteria
#
max_smajax=3000.0
residual_over_sigma_max=3.
max_obs_net_prob=5
req_num_of_defining_detections=2
weight_threshold=3.55
primary_time_weight=1.0
hydro_time_weight=1.2
infra_time_weight=1.
secondary_time_weight=0.7
array_azimuth_weight=0.4
array_slow_weight=0.4
3comp_slow_weight=0.2
3comp_azimuth_weight=0.2
hydro_azimuth_weight=0.6
hydro_slow_weight=0.
infra_azimuth_weight=0.4
infra_slow_weight=0.4
#
# Arrival quality test parameters
#
ar_qual_alpha=0.75
ar_qual_beta=0.1
ar_qual_alpha_hydro=0.
ar_qual_beta_hydro=0.46
```

```
ar_qual_alpha_infra=0.0008
ar_qual_beta_infra=0.35
ar_qual_gamma=0.0
ar_qual_thresh=1.35
#
# Cluster analysis parameters
#
cluster_min_ndef=6
cluster_min_pct_overlap=0.8
#
# Phase prediction module parameters
#
predicted_phases="T,S,PcP,ScP,PKPab,PKPbc,PP,ScS,Lg,Sn,Rg,\
  PKKPab,PKKPbc,PKKPdf,PKP2ab,PKP2bc,PKP2df,SKPab,SKPdf,SKKPdf,\
  SKKPbc,SP,PKiKP"
#
# Association-based conflict resolution parameters
#
master_tradeoff_weight=0.2
event_likelihood_weight=1.0
dissolved_event_weight=0.0
ndef_not_likely_to_dissolve_event=6
ndef_which_will_dissolve_event=3
ndef_weight=1.0
smajax_weight=0.05
dnear_weight=0.0
probdet_weight=0.0
hydro_H_weight=1.
hydro_S_weight=0.
infra_I_weight=1.
infra_S_weight=0.
ndef_no_confidence_bound=3
ndef_high_confidence_bound=10
hydro_ndef_no_confidence_bound=0
hydro_ndef_high_confidence_bound=4
infra_ndef_no_confidence_bound=0
infra_ndef_high_confidence_bound=4
seismic_ndef_no_confidence_bound=0
seismic_ndef_high_confidence_bound=20
smajax_no_confidence_bound=3000.0
```

```
smajax_high_confidence_bound=10.0
dnear_no_confidence_bound=90.0
dnear_high_confidence_bound=10.0
probdet_no_confidence_bound=3.0
probdet_high_confidence_bound=1.0
#
# Magnitude difference parameters
#
# Default maximum magnitude difference
#
default_max_magnitude_diff=2.
#max_magnitude_diff=NORES,2.,ARCES,2.,FINES,2.,PDY,2.,MJAR,2.,KSAR,2.
max_uncertainty_magnitude_diff=.6
min_sta_magnitude_diff=2
#
# shear-phase test parameters
#
max_dist_shear_test=10.
max_ndef_shear_test=2
shear_phases="S,Sn,Lg"
#
restricted_shear_phases="S,Sn"
s_constraint_3c_snr_thresh=-1.
s_constraint_3c_hvrat_thresh=-1.
s_constraint_3c_rect_thresh=-1.
s_constraint_ar_apma_required=0
s_constraint_ar_snr_thresh=-1.
s_constraint_ar_hvrat_thresh=-1.
s_constraint_ar_rect_thresh=-1.
#
equivalence_list="(P Pn Pg PKP),(Pn Pg P),(Pg Pn P),(PKP P),\
  (Sn Lg S),(S Lg Sn),(Lg S Sn)"
phase_ordering_list="(Pn Pg Sn Lg Rg),(P S),(P PcP),(P PP),\
  (S ScS),(S SS)"
#
respect_time_limit
```

The example that follows shows the SEL2 pipeline-specific parameter file. This file includes a reference to the `GA.par` parameter file from the directory *<dir>*/`app_config/automatic/GA`.

```
##############################################################
# SEL2 pipeline parameters common to GAassoc
# and GAconflict.
#############################################################
#
par=$(IMSPAR)
par=$(AUTOMATIC)
par=$(PARDIR)/GA/GA.par
#
# Define environment variables for database tables
# to be shared by GAassoc, GAconflict
#
# Define Arrival and amplitude tables
#
GA_ARRIVAL=arrival
GA_AMPLITUDE=amplitude
GA_APMA=apma
GA_HYDRO_ASSOC=hydro_assoc
GA_HYDRO_ARR_GROUP=hydro_arr_group
#
# Define static tables
#
GA_SITE=site
GA_SITEAUX=siteaux
GA_AFFILIATION=affiliation
#
# Define stapro tables.
#
GA_STAPRO_ORIGIN=origin_stapro
GA_STAPRO_ORIGERR=origerr_stapro
GA_STAPRO_ASSOC=assoc_stapro
#
# Tables to be written by GAassoc and read by GAconflict
#
GA_TEMP_ORIGIN=origin_temp_ga
GA_TEMP_ORIGERR=origerr_temp_ga
```

```
GA_TEMP_ASSOC=assoc_temp_ga
#
# Tables to be read and written to by GAconflict(previous tables)
# DIFFERENT SETTING comparing to Beta pipeline
#
GA_ORIGIN=origin
GA_ORIGERR=origerr
GA_ASSOC=assoc
#
# Previous tables
#
PRE_ORIGIN=sel1.origin
PRE_ORIGERR=sel1.origerr
PRE_ASSOC=sel1.assoc
#
# Tables to be written to by GAconflict
#
GA_EVENT=event
GA_STAMAG=stamag
GA_NETMAG=netmag
#
# Tag table
#
GA_TAG_TABLE=ga_tag
#
# database account
#
account=$(SEL2DB)
#
# Networks of stations. Include all seismic
#
net=CUR_PRI,CUR_AUX,CUR_HYD,AA_INF
net_aux=CUR_AUX
hydro_net=CUR_HYD
infra_net=AA_INF
netmag_network=SEISMIC
#
# Phases to be used in association loop
#
phases="P,PKP,Pn,Pg,H,I"
```

GA_DBI_sel1.par, GA_DBI_sel2.par, GA_DBI_sel3.par

Directory: *<dir>*`/app_config/automatic/GA_DBI`

As reflected in their names, these files are specific to each of the SEL1, SEL2, and SEL3 pipelines and refer to the file in the `/GA` directory that is also specific to the pipeline.

The example that follows shows the `GA_DBI_sel2.par` parameter file. It includes a reference to the `GAsel2.par` file from directory *<dir>*`/app_config/automatic/GA`.

```
##########################################################
#
#
# Parameters for GA_DBI. SEL2 pipeline
#
#
##########################################################
#
par=$(IMSPAR)
par=$(AUTOMATIC)
par=$(PARDIR)/GA/GAsel2.par
hydro_snr_thresh=0.1
hydro_max_H_per_hour=5
#
# Input Tables
#
in-arrival-table=$(GA_ARRIVAL)
in-site-table=$(GA_SITE)
in-siteaux-table=$(GA_SITEAUX)
in-affiliation-table=$(GA_AFFILIATION)
in-amplitude-table=$(GA_AMPLITUDE)
in-apma-table=$(GA_APMA)
hydro-assoc-table=$(GA_HYDRO_ASSOC)
hydro-arr-group-table=$(GA_HYDRO_ARR_GROUP)
#
in-origin-table=$(GA_STAPRO_ORIGIN)
origin-sp-table=$(GA_STAPRO_ORIGIN)
```

```
in-origerr-table=$(GA_STAPRO_ORIGERR)
in-assoc-table=$(GA_STAPRO_ASSOC)
assoc-sp-table=$(GA_STAPRO_ASSOC)
#
ga-tag-table=$(GA_TAG_TABLE)
#
# Output Tables
#
out-origin-table=$(GA_TEMP_ORIGIN)
out-origerr-table=$(GA_TEMP_ORIGERR)
out-assoc-table=$(GA_TEMP_ASSOC)
#
# Previous tables
#
pre-origin-table=$(GA_ORIGIN)
pre-origerr-table=$(GA_ORIGERR)
pre-assoc-table=$(GA_ASSOC)
#
# Final tables
#
final-origin-table=$(GA_ORIGIN)
final-assoc-table=$(GA_ASSOC)
#
```

GAassoc_sel1.par, GAassoc_sel2.par, GAassoc_sel3.par

Directory: *<dir>*`/app_config/automatic/GAassoc`

As reflected in their names, these files are specific to the SEL1, SEL2, and SEL3 pipelines, and each of them refers to the file in the `/GA` directory that is also specific to the pipeline.

The example that follows shows the `GAassoc_sel2.par` file. The parameter file includes a reference to the `GAsel2.par` file from directory *<dir>*`/app_config/automatic/GA`.

```
##############################################################
#
#
#      Parameters for GAassoc. SEL2 pipeline
#
#
##############################################################
#
par=$(IMSPAR)
par=$(AUTOMATIC)
par=$(PARDIR)/GA/GAsel2.par
#
# Input Tables
#
in-arrival-table=$(GA_ARRIVAL)
in-site-table=$(GA_SITE)
in-siteaux-table=$(GA_SITEAUX)
in-affiliation-table=$(GA_AFFILIATION)
in-amplitude-table=$(GA_AMPLITUDE)
in-apma-table=$(GA_APMA)
hydro-assoc-table=$(GA_HYDRO_ASSOC)
hydro-arr-group-table=$(GA_HYDRO_ARR_GROUP)
#
in-origin-table=$(GA_STAPRO_ORIGIN)
origin-sp-table=$(GA_STAPRO_ORIGIN)
in-origerr-table=$(GA_STAPRO_ORIGERR)
in-assoc-table=$(GA_STAPRO_ASSOC)
assoc-sp-table=$(GA_STAPRO_ASSOC)
#
ga-tag-table=$(GA_TAG_TABLE)
#
# Output Tables
#
out-origin-table=$(GA_TEMP_ORIGIN)
out-origerr-table=$(GA_TEMP_ORIGERR)
out-assoc-table=$(GA_TEMP_ASSOC)
#
# Previous tables
#
pre-origin-table=$(GA_ORIGIN)
```

```
pre-origerr-table=$(GA_ORIGERR)
pre-assoc-table=$(GA_ASSOC)
#
# Final tables
#
final-origin-table=$(GA_ORIGIN)
final-assoc-table=$(GA_ASSOC)
#
```

GAconflict_sel1.par, GAconflict_sel2.par, GAconflict_sel3.par

Directory: *<dir>*`/app_config/automatic/GAconflict`

As indicated by their names, these files are specific to the SEL1, SEL2, and SEL3 pipelines, and each of them refers to the file in the `/GA` directory that is also specific to the pipeline.

The following example shows the `GAconflict_sel2.par` file:

```
##########################################################
#
#
#      Parameters for GAconflict. SEL2 pipeline.
#
#
##########################################################
par=$(IMSPAR)
par=$(AUTOMATIC)
par=$(PARDIR)/GA/GAsel2.par
#
# Table names.
#
#
in-assoc-table=$(GA_TEMP_ASSOC)
in-origin-table=$(GA_TEMP_ORIGIN)
in-origerr-table=$(GA_TEMP_ORIGERR)
origin-sp-table=$(GA_STAPRO_ORIGIN)
assoc-sp-table=$(GA_STAPRO_ASSOC)
hydro-assoc-table=$(GA_HYDRO_ASSOC)
```

```
hydro-arr-group-table=$(GA_HYDRO_ARR_GROUP)
#
event-table=$(GA_EVENT)
netmag-table=$(GA_NETMAG)
stamag-table=$(GA_STAMAG)
in-site-table=$(GA_SITE)
in-siteaux-table=$(GA_SITEAUX)
in-affiliation-table=$(GA_AFFILIATION)
in-arrival-table=$(GA_ARRIVAL)
in-amplitude-table=$(GA_AMPLITUDE)
in-apma-table=$(GA_APMA)
ga-tag-table=$(GA_TAG_TABLE)
#
out-origin-table=$(GA_ORIGIN)
out-origerr-table=$(GA_ORIGERR)
out-assoc-table=$(GA_ASSOC)
#
# Previous tables
#
pre-origin-table=$(PRE_ORIGIN)
pre-origerr-table=$(PRE_ORIGERR)
pre-assoc-table=$(PRE_ASSOC)
#
# Final tables
#
final-origin-table=$(GA_ORIGIN)
final-assoc-table=$(GA_ASSOC)
#
```

GAcons.par

Directory: *<dir>*`/app_config/automatic/GAcons`

The following example shows a *GAcons* parameter file:

```
##############################################################
#
# Parameters for GAcons seismic grid generation
#
##############################################################
exit-ga-no-arrivals=10
exit-gdi-open-failure=11
exit-gdi-submit-failure=12
exit-ga-memory-allocation-failure=13
exit-ga-error=14
#
par=$(IMSPAR)
par=$(AUTOMATIC)
vendor="oracle"
database=""
account=$(SEL1DB)
maxrecs=10000
in-site-table=site
in-siteaux-table=siteaux
in-affiliation-table=affiliation
channels="cb,bz,sz,ed,ez,is"
grid_spacing=3.0
num_sects=1
#net="AA_IDC"
net="CUR_PRI,CUR_HYD,AA_INF"
hydro_net=CUR_HYD
hydro_types=hy
infra_net=AA_INF
infra_types=is
#
# jdate is the Julian date of the day for which the
# grid is to be computed. Reading of the site table
# takes into account the ondate and offdate
# of individual stations.
#
jdate=2000297
percentage_of_time_event_is_detected=1
num_stat_detecting_event=2
num_events_for_simulation=200
min_mag=3.0
```

```
max_mag=5.0
vmodel_spec_file=$(VMSF)
magnitude_spec_file=$(MDF)
list_of_magtypes="mb,ml"
transmission_loss_spec_file=$(TLSF)
mag_atten_dir_prefix=$(ATTEN_PREFIX)
blockage_spec_dir=$(STATICDIR)/BLK_OSO
phases="Pg,Pn,P,Pdiff,PKP,H,I"
hydro_phases="H,T"
infra_phases="I"
output_path=$(GA_SITE_PATH)/create
output_prefix="AA_IDC"
grid_prefix="AA_IDC"
atten_file=$(ATTEN_DIR)/slowamp.P
event_file=$(GA_SITE_PATH)/1980-1993.pde_depths
dist_depth_range_file=$(GA_SITE_PATH)/GA_dist_depth_ranges
min_num_events_per_10sq_deg=1.0
depth_points="65.0,130.0,240.0,400.0,600.0"
depth_widths="32.0, 40.0, 80.0,100.0,100.0"
```

Configuration Data Files

The configuration data files required for running the GA applications are the parameter files for each of the applications, the data files required as input, and the two binary grid files generated by *GAcons* and used by *GAassoc* and *GAconflict*. The <*dir*> directory is set using a UNIX variable. The following data files are required as input to the operational executables:

- File <*dir*>`/earth_specs/GA/GA_dist_depth_ranges` contains the distance and depth domains of definition of seismo-acoustic phases. The information is specified on one line for each phase, with columns:

  <Phaseid> <min-dist(deg.)> <max-dist(deg.)> <min-depth(km)> <max-depth(km)>

- File <*dir*>`/earth_specs/MAG/slowamp.P` contains attenuation data used in the computation of the probability of detection. The format of the file is:

  <distance (deg.)> <slowness (s/deg.)> <amplitude_correction> <s.d. of correction>

- Blockage files for the hydroacoustic stations are located in <*dir*>`/earth_specs/BLK_OSO`.
- Travel-time configuration data are located in <*dir*>`/earth_specs/TT`, which includes the *vmsf* file to configure the velocity model to be used in the travel-time handling system.
- Magnitude configuration data are located in <*dir*>`/earth_specs/MAG`, which includes the *tlsf* and *mdf* files used to configure the magnitude handling system.
- File <*dir*>`/earth_specs/GA/1980-1993.pde_depth` contains deep seismicity data for the period 1980 to 1993. The file contains the seismicity information in the format:

  <latitude> <longitude> <depth> <mb>

The binary grid files generated by *GAcons* and used by *GAassoc* and *GAconflict* are located in the directory <*dir*>`/earth_specs/GA`. An example of files in that directory are the file `AA_IDC.sector.-180deg.to.180deg`, which contains the propagation knowledge, and the file `AA_IDC.grid.3.all_sectors`, which contains the geographical information about the grid.

DATABASE

This section describes database elements required for operation of this software component, including accounts, tables, and initialization of the **lastid** table.

Accounts

The software produces the automatic bulletins in the SEL1, SEL2, and SEL3 database accounts. Establishing these accounts, with the required tables, is a prerequisite for operating this software. The IDCX account is also required to operate the software and to read the dynamic data from tables IDCX**.arrival,** IDCX**.amplitude,** IDCX**.apma,** IDCX**.hydro_assoc**, and IDCX**.hydro_arr_group**. The STATIC account is required to access static information in tables STATIC**.site,** STATIC**.affiliation**, and STATIC**.siteaux**.

Tables

In each of the SEL1, SEL2, and SEL3 accounts, the following tables are required to operate GA: **assoc**, **origin**, **origerr**, **event**, **netmag**, **stamag**, **assoc_temp_ga**, **origin_temp_GA**, and **origerr_temp_ga**.

In addition, SEL1**.ga_tag** contains information about the arrivals, such as the tags identifying auxiliary stations arrivals or restricted use arrivals. Each of the SEL2 and SEL3 accounts contain a synonym to SEL1**.ga_tag**. Insert and update privileges are granted on this table to the SEL2 and SEL3 accounts.

Synonyms to tables in other accounts must be established to operate the system. From the STATIC database account, synonyms to the following tables are needed with a select privilege: STATIC**.affiliation**, STATIC**.site**, `STATIC`**.siteaux.**

From the IDCX database account, synonyms to the following tables are needed with a select privilege: IDCX**.arrival**, IDCX**.amplitude**, IDCX**.apma**, IDCX**.lastid**, IDCX**.assoc_stapro**, IDCX**.origerr_stapro**, IDCX**.origin_stapro**, IDCX**.hydro_assoc**, and IDCX**.hydro_arr_group.**

Initialization of lastid

GA generates the automatic bulletins and calculates magnitudes. For this purpose, GA increments the values for *orid* and *magid* in the **lastid** table when creating new origins or calculating new magnitudes. The value for **origin.***evid* is set to the same value as **origin.***orid*, as obtained from the **lastid** table. When starting a completely new instance of the automatic association system, the values of the *orid* and *magid* rows in the **lastid** table may be set to a value of `1`. When pre-existing event hypotheses are present in the database accounts, the values for *orid* and *magid* should be set to numbers larger than the maximum of these values over all event hypotheses in the database to avoid errors arising from non-uniqueness.

INITIATING OPERATIONS

In a usual operational context, GA is initialized as a part of the automatic processing pipelines SEL1, SEL2, and SEL3. This initialization is performed using the *tuxpad* application (see "Software Startup" on page 12). A detailed explanation of how to start the pipelines is given in [IDC6.2.1]. The *tuxpad* GUI is documented in [IDC6.5.2Rev0.1].

VALIDATING INSTALLATION

Use the following steps to validate the operation of GA:

- Monitor the processing using the *WorkFlow* tool and ensure that intervals are being properly queued and processed with no failures.
- Perform database queries to ensure that the bulletins are being produced.
- Monitor the log files to ensure that the software operates as intended.

Refer to "Monitoring" on page 56 for more details on validating installation and monitoring this software.

References

The following sources supplement or are referenced in document:

[Bac93]	Bache, T. C., Bratt, S. R., Swanger, H., Beall, G., and Dashiell, F. K., "Knowledge-Based Interpretation of Seismic Data in the Intelligent Monitoring System," *Bulletin of the Seismological Society of America,* Volume 83, pp. 1507–1526, 1993.
[IDC5.1.1Rev2]	Science Applications International Corporation, Veridian Pacific-Sierra Research, *Database Schema, Revision 2*, SAIC-00/3057, PSR-00/TN2830, 2000.
[IDC5.1.3]	Science Applications International Corporation, Pacific-Sierra Research, Inc., *Configuration of PIDC Databases,* SAIC-99/3019, PSR-99/TN1114, 1999.
[IDC5.2.1]	Science Applications International Corporation, *IDC Processing of Seismic, Hydroacoustic, and Infrasonic Data*, SAIC-99/3023, 1999.
[IDC6.2.1]	Science Applications International Corporation, *Release 2 Operations and Maintenance–Seismic, Hydroacoustic, and Infrasonic System,* SAIC-00/3000, 2000.
[IDC6.5.2Rev0.1]	Science Applications International Corporation, *Distributed Application Control System (DACS) Software User Manual, Revision 0.1*, SAIC-00/3038, 2000.
[IDC7.1.4]	Science Applications International Corporation, *Global Association (GA) Subsystem,* SAIC-01/3009, 2001.

[Kat98] Katz, C. N., Brown, D. J., Gault, A. K., LeBras, R., and Wang, J., P*IDC 6.0: Implementation of Infrasonic Processing in PIDC*, CCB-PRO-98/11, 1998.

[LeB96] LeBras, R., *User Manual for the Global Association Subsystem*, Science Applications International Corporation, SAIC-96/1128, 1996.

Appendix: GAgrid User Manual

This Appendix describes *GAgrid* capabilities for visualizing the grid cells and their outlines, the station locations, and paths from grid cells to station locations. The following topics are covered:

Appendix: GAgrid User Manual

INTRODUCTION

GAgrid is a GUI program that graphically displays the grid file built by the program *GAcons*. The grid file is the knowledge-base used by *GAassoc* to form event hypotheses in the initial phase of the automatic association process, and as such is an important component of the automatic association process.

The binary grid file consists of the following information:

- the geographic location information for the grid points
- the list of stations in the S/H/I networks
- the travel time and slowness information for a list of S/H/I phases for each path between the station and each grid cell

This Appendix describes *GAgrid* capabilities for visualizing the grid cells and their outlines, the station locations, and paths from grid cells to station locations. All graphic displays are superimposed on a global map background with coastal outlines and political boundaries. The grid content may be examined alphanumerically or graphically.

USING GAGRID FEATURES

Graphical Displays

GAgrid is a GUI program for visualizing the data built by *GAcons* and organized in a binary file. The grid file contains network-specific information about propagation from a set of cells to stations in the network. The grid cells may cover the entire globe or be limited to a region. Overlapping circular grid cells provide complete global coverage, including depth cells in areas where deep seismicity is known to occur. The algorithm used by *GAcons* to build the grid is explained in detail in [IDC5.2.1]. One of GA's functions is to establish a list of stations for each cell that can potentially see the first arrival from an event within that cell. These stations are referred to as the "First Stations."

The user interface's main graphic window presents a map of the world and the outline of the grid cells generated by *GAcons*. In the initial display, the grid cells are represented as small circles centered on the grid point to optimize the use of display space. The subsequent displays show the cells at their actual size. Stations are represented as triangles. Initially, all surface cells are presented. The following sections describe options for changing the display.

Zooming-Unzooming Graphical Displays

The initial display of the grid cells can be quite crowded, especially for surface cells, and it is useful to zoom to regions of interest.

To zoom:

1. Press the middle mouse button.
2. Drag on the display.

   A rectangle with the same aspect ratio as the overall display appears, representing the area to be zoomed on.

The size of the rectangle can be changed until the desired area is defined. To unzoom, click on the middle mouse button.

Displaying Grid Cells at Depth

For the surface of the earth, every location is included in the grid, but depth cells below the surface locations are included only if the historical event density exceeds some threshold. Figure A-1 shows an example of the cells with centers at 65 km depth.

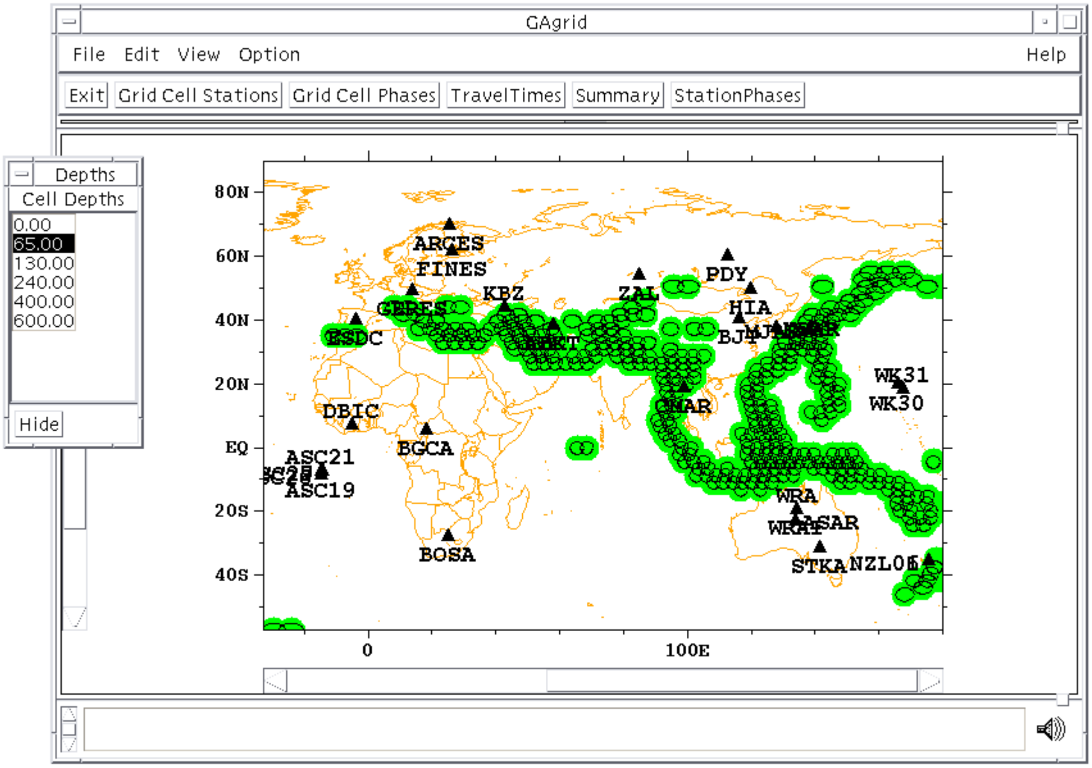


FIGURE A-1. GAGRID DISPLAY FOR DEPTH OF 65 KM

To select a particular depth to be displayed:

1. Choose View>Depths.
2. Select a particular depth to be displayed.

   The grid cells at depth are displayed.

Displaying Great Circle Paths to First Arrival Stations

You can display all grid cells, selected grid cells, or none of the grid cells.

To display great circle paths to the first-arrival stations:

1. Select a cell by clicking on its graphical representation on the map. To select several cells in addition to the primary grid cell, press <cntrl> on the keyboard while clicking the left mouse button.
2. Choose View>Display Grid Cells.
3. Choose the All, Selected, or None option.

   The selected cell is highlighted.
4. Choose View>Display Stations>All to display all of the stations, or choose the First Stations option for just the first arrival station for that cell.

   When you choose the All option, all stations are displayed. When you choose the First Stations option, the great circle paths from the center of the cell to each of the "First Stations" are displayed. When you choose the None option, no stations are displayed.

Choosing Map Projections and Displaying Lat/Lon Lines

You can choose from two different map projections, linear cylindrical or orthographic, and you can also draw latitude and longitude lines.

To select a projection:

1. Choose View>Projection and select from the two projection options: Linear cylindrical and Orthographic.

To draw the latitude and longitude lines:

1. Choose View>Lat/Lon grid (Figure A-2).

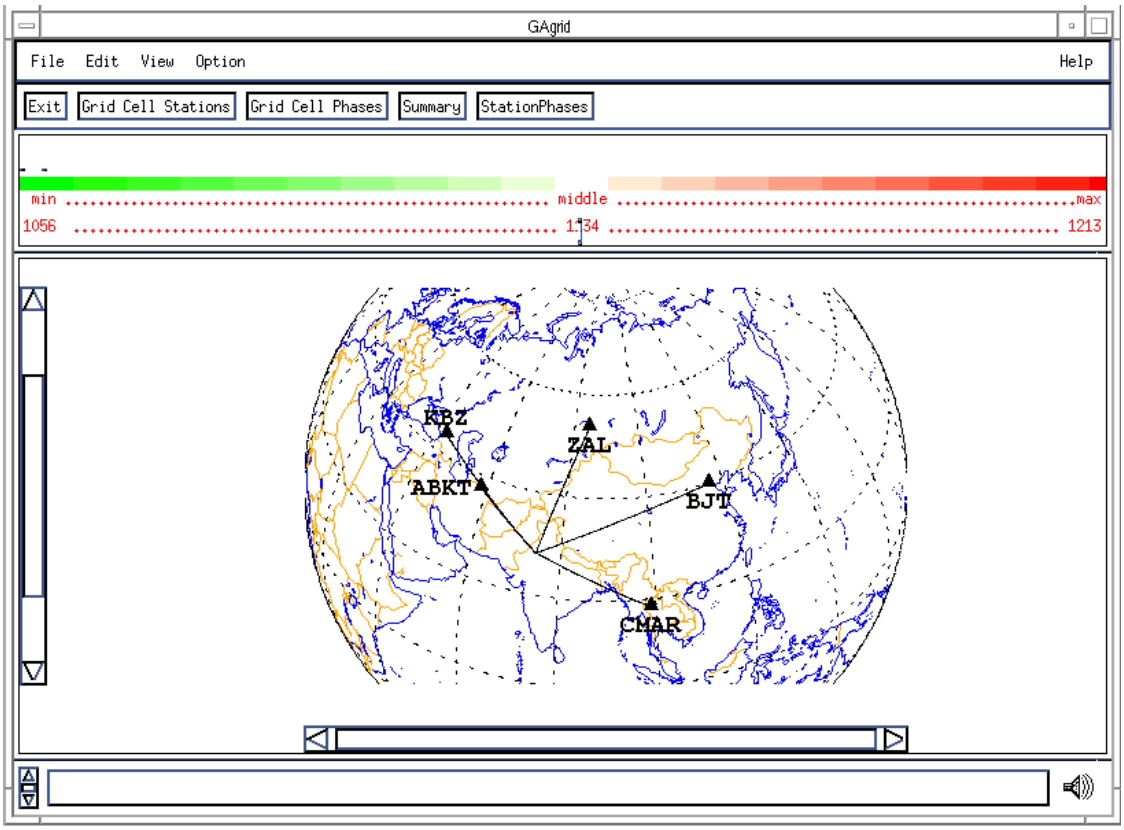


FIGURE A-2. GREAT CIRCLE PATHS ON ORTHOGRAPHIC PROJECTION WITH LATITUDE AND LONGITUDE LINES

Using Alphanumeric Displays

You can display the information contained in the file for each cell and station-cell pair in a table.

To use the alphanumeric displays:

1. Select a cell by clicking on it. You must select at least one cell (Figure A-3).

   Two types of alphanumeric displays are available. The first type shows information about the geometry between cell and stations with no propagation (phase) information. The second type shows propagation (phase) information between the selected cell and all stations in the network.

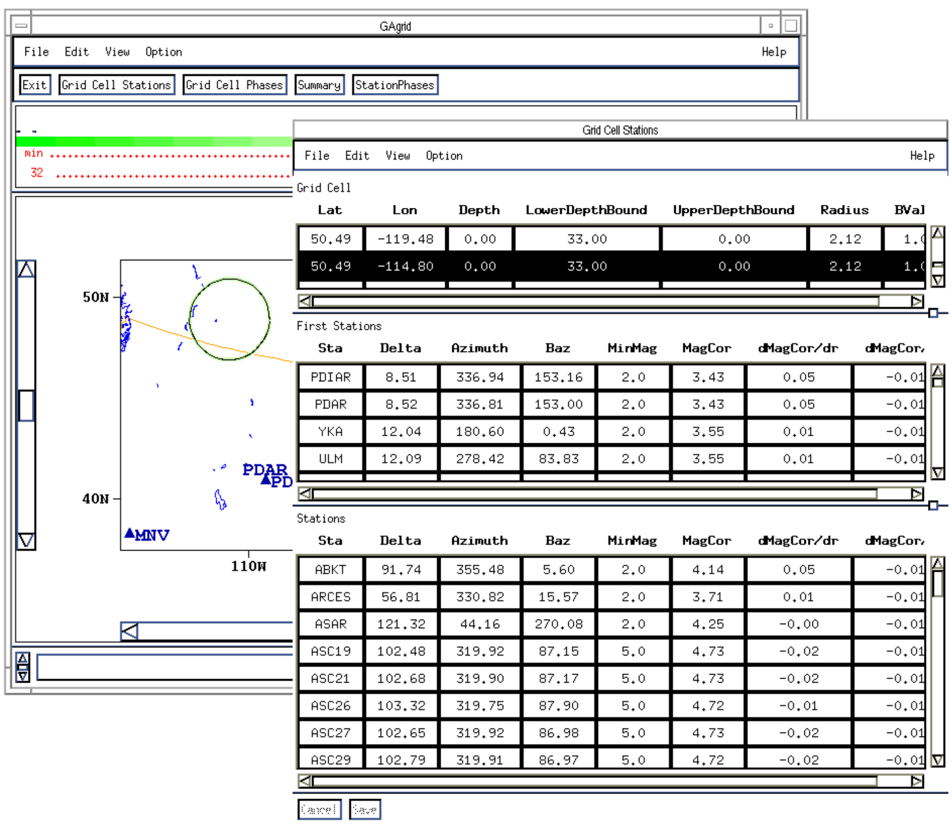


FIGURE A-3. ALPHANUMERIC DISPLAY OF CELL GEOMETRY

2. To select the first type of alphanumeric display, click the Grid Cell Stations button on the button bar.

   A three-section window displays the phase-independent information for the selected cells.

   The first part of the display shows the Latitude, Longitude, Depth, LowerDepthBound, UpperDepthBound, cell Radius, and BVal (b value). When several cells are selected, the list displays these values for each of the selected cells. One of the rows on the display is highlighted (the row for the primary cell), and the information for the corresponding cell is displayed on the second and third part of this display and on the propagation information display if the latter has been selected.

   The second part of the display lists all potential first stations for the highlighted cell, ordered by distance. If present, the hydroacoustic stations are listed first because they are always considered as potential driver stations. The stations list contains the distances to the center of the cell, the azimuths and back-azimuths to the center of the cell, the minimum magnitudes for an event to be detected at the stations, the magnitude corrections at the center of the cell, and derivatives of the magnitude corrections with respect to the distance and depth.

   The third part of the display lists the same information as listed by the First Station list for all stations (in alphabetical order).

3. To select the second type of alphanumeric display (propagation [phase] information between the selected cell and all stations in the network), click the Grid Cell Phases button on the button bar.

   A graphics window appears (Figure A-4). This window shows the following phase-dependent attributes for each cell-station combination:

   – station name
   – phase name
   – travel time at the center of the grid (*TTime*)
   – minimum travel time (*Min_TTime*)
   – maximum travel time (*Max_TTime*)

– travel time derivative at the center of the grid with respect to distance (*dTTime/dr*)

– travel time derivative with respect to depth (*dTTime/dz*)

– maximum slowness difference between the center of the cell and any point within the cell (*DelCell*)

These attributes are important for the initial association step within the program *GAassoc*. A description of the association process and the use of these attributes within the process is provided in [IDC5.2.1].

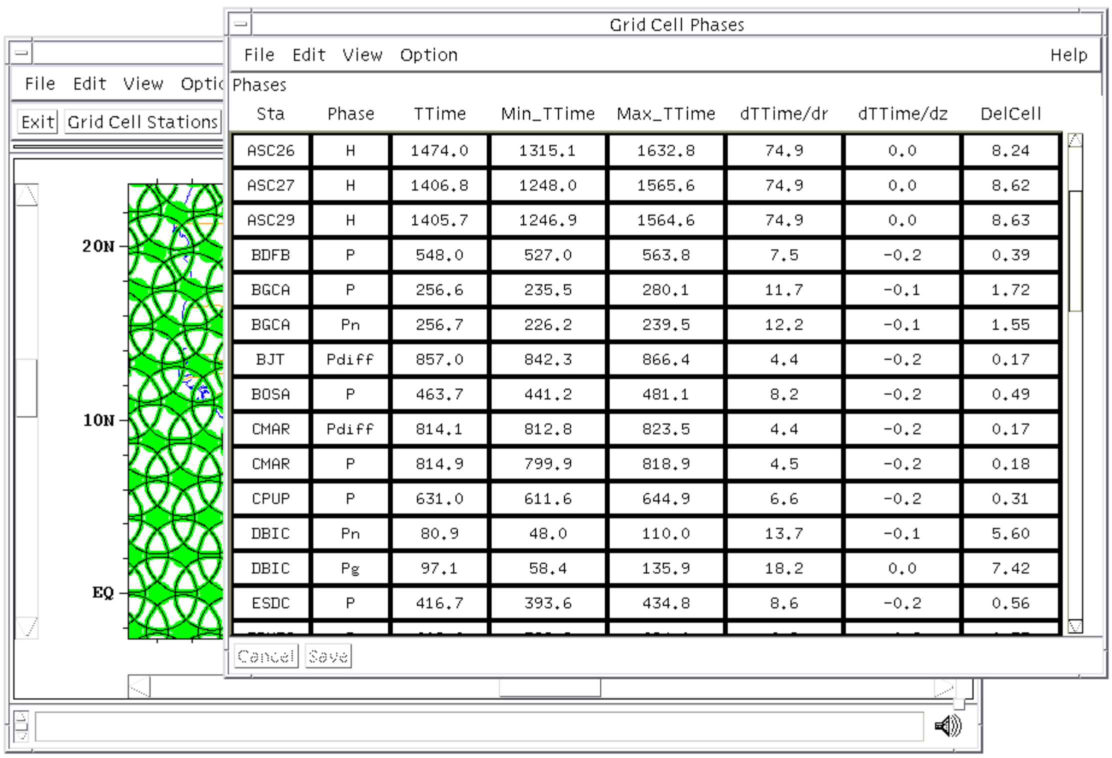


FIGURE A-4. STATION-PHASE INFORMATION AT SELECTED GRID CELL

Displaying Summary of Station Network

To display a list of all stations included in the grid file, click the Summary button (Figure A-5).

Geographically Displaying Attributes

A large quantity of information is contained within each cell, and although the alphanumeric displays are useful for checking the content of any one cell at the time, you may find it useful to show attributes as geographical displays.

To show attributes as geographical displays:

1. Choose View>Travel times for Station/Phase for the list of phases present at a station.
2. Select a particular item from the list (as shown in Figure A-6 for station ILAR and phase Pn, and in Figure A-7 on page A13 for station ASC26 and phase H).

   The cells that include this particular station-phase combination are displayed, color-coded by the value of the *travel-time* attribute at the center of the cell.

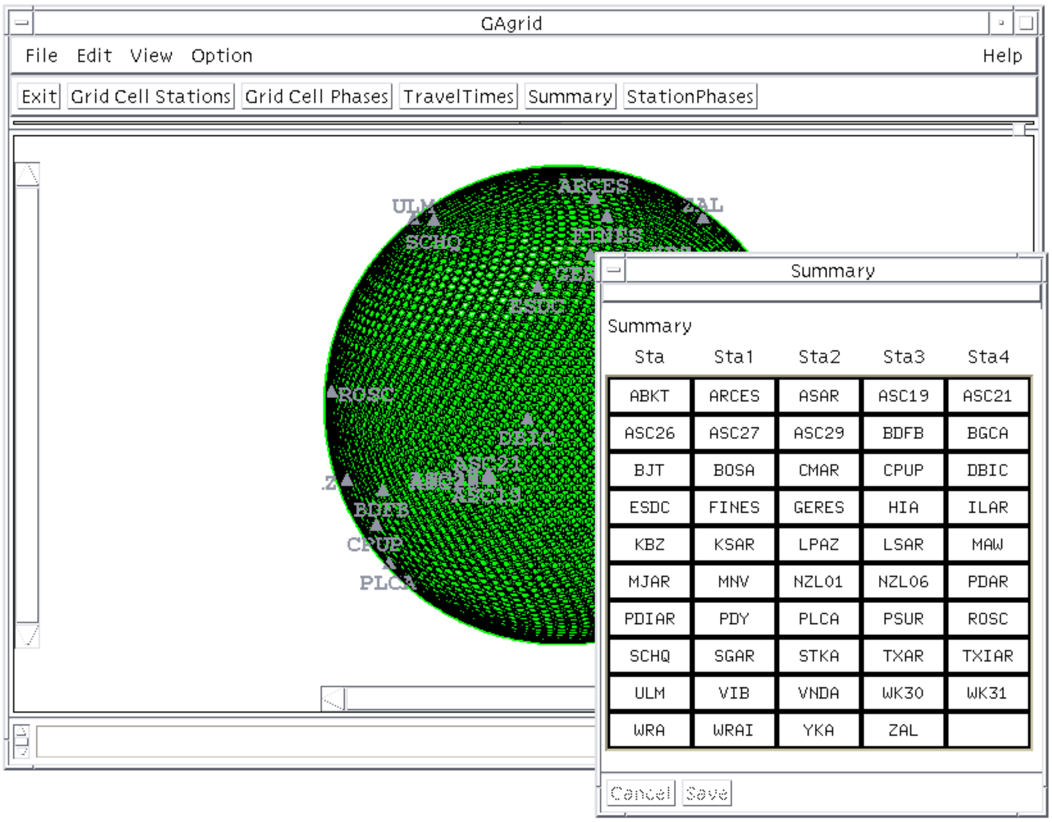


FIGURE A-5. SUMMARY DISPLAY OF STATIONS IN GRID FILE

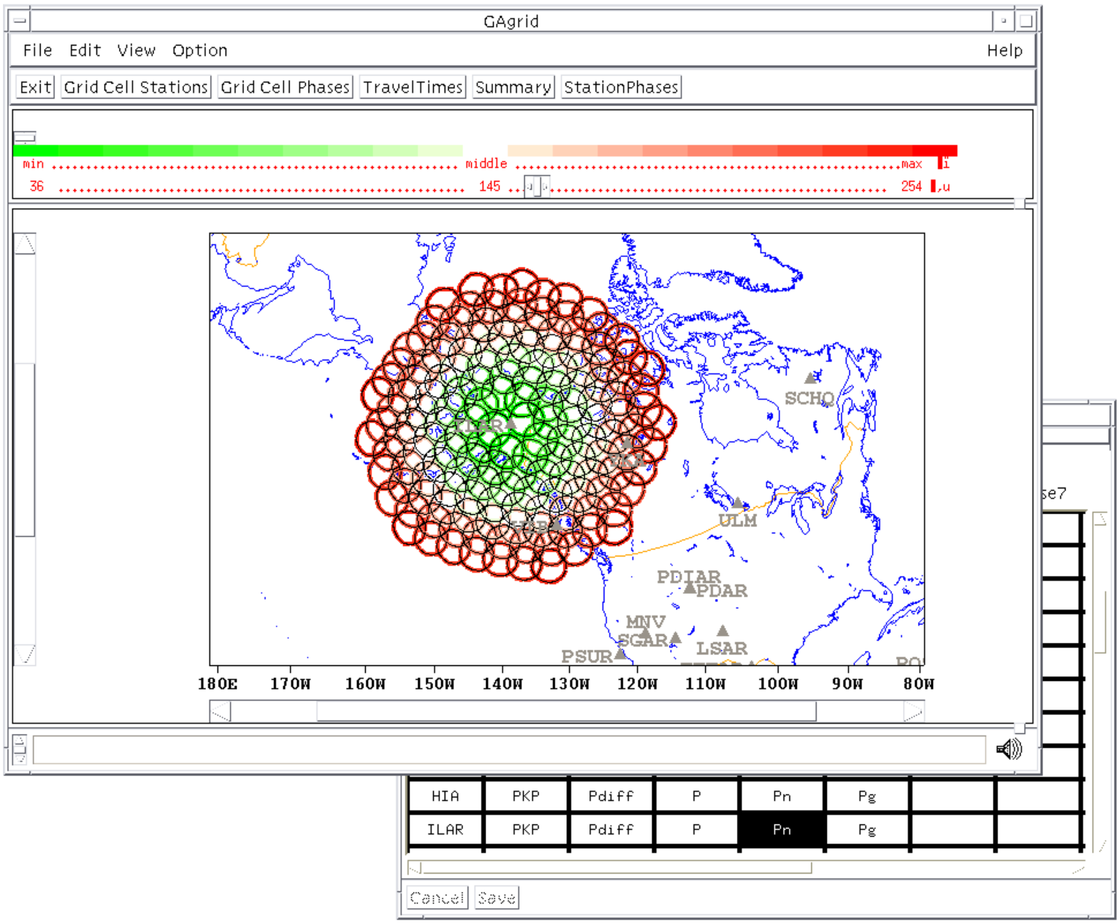


FIGURE A-6. COLOR-CODED TRAVEL TIME FOR ILAR PN PHASE

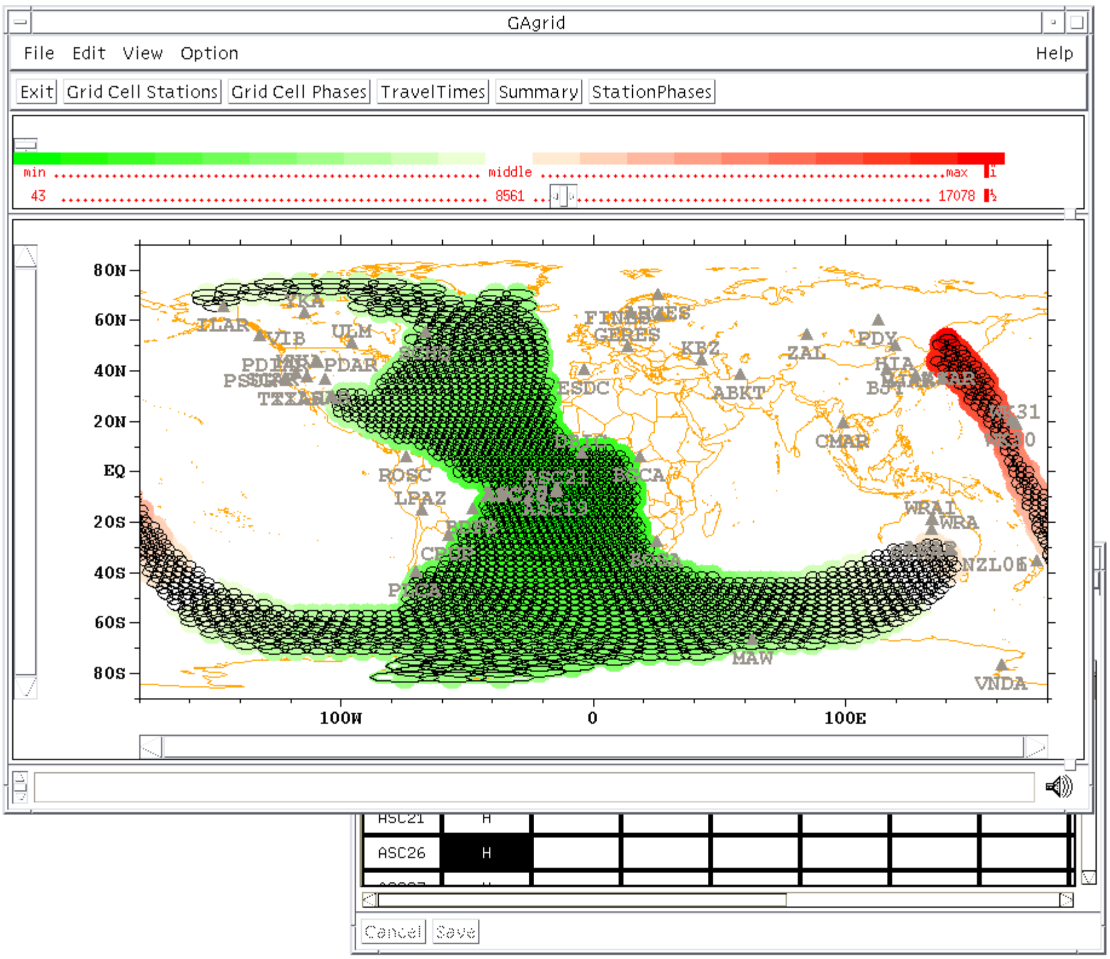


FIGURE A-7. COLOR-CODED TRAVEL TIME FOR ASC26 H PHASE

USING GAgrid FUNCTIONS

Mouse Options

Table A-1 describes the *GAgrid* functions available through the mouse buttons.

TABLE A-1: MOUSE BUTTON FUNCTIONS

Action	Function
left click	Select primary grid cell (highlighted on the alphanumerical stations display).
cntrl left click	Select grid cell.
middle drag	Zoom on selected rectangular area.
middle click	Unzoom.
shift middle click	Repeat zoom.
cntrl middle click	Rotate.

Table A-2 describes the *GAgrid* functions available through the keyboard.

TABLE A-2: KEYBOARD FUNCTIONS

Action	Function
h	Zoom horizontally.
v	Zoom vertically.
shift-h	Unzoom horizontally.
shift-v	Unzoom vertically.
c	Add crossbar.
d	Delete nearest crossbar or double lines.
shift-d	Delete all cursors.
z	Zoom in 20 percent.
shift-z	Zoom out 20 percent

TABLE A-2: KEYBOARD FUNCTIONS (CONTINUED)

Action	Function
f, F	Scroll down.
b, B	Scroll up.
r, R	Scroll right.
t, T	Scroll left.

User Interface Organization

This section explains the layout of the title bar menu buttons.

- File>Exit

  Exits the *GAgrid* program.

- Edit

  (This option is not yet operational.)

- View>Depths

  Displays a scrollable list of depths. When a depth is selected, only grid cells at that depth are displayed on the map.

- View>StationCells

  Provides a list of the phases available per station. Each row of the display corresponds to a station, and phases are listed in the columns.

- View>Display Grid Cells>All

  Displays all grid cell perimeters on the current map.

- View>Display Grid Cells>Selected

  Displays only previously selected cells on the current map.

- View>Display Grid Cells>None

  No cell is displayed on the current map.

- View>Display Stations>All

  Displays all stations included within the grid file on the map.

- View>Display Stations>First Stations

  Displays all first stations for the selected cell, as well as the great circle path joining the center of the cell to these stations.

- View>Display Stations>None

  Removes all stations from the display.

- View>Lat/lon Grid

  This option is a toggle button to add or remove the latitude and longitude grid to the map display.

- View>Projection>Linear Cylindrical

  Displays linear cylindrical map projection.

- View>Projection>Orthographic

  Displays orthographic map projection.

- Option

  (The Option box is not yet operational.)

- Help>Mouse Buttons

  Explains how to use the mouse and keyboard.

- Help>Options

  Explains how to use a few menu options.

BUTTON BAR

The following buttons are provided as short cuts for the pull-down menu options.

Exit

Clicking the Exit button is equivalent to choosing File>Exit from the menu bar.

Grid Cell Stations

Clicking the Grid Cell Stations button displays information pertaining to the selected cell(s) in an alphanumeric display window. A description of that window is given in "Using Alphanumeric Displays" on page A6, and the display is shown in Figure A-3 on page A7.

Grid Cell Phases

Clicking the Grid Cell Phases button displays cell-station-phase attributes in an alphanumeric display window. A description of that window is provided in "Using Alphanumeric Displays" on page A6, and the display is shown on Figure A-4 on page A9.

Summary

Clicking the Summary button displays all the stations present in the grid file in an alphanumeric display window.

StationPhases

Clicking the StationPhases button is equivalent to choosing View>StationCells from the menu bar.

Glossary

Symbols

2-D

Two-dimensional.

3-C

Three-component.

A

amplitude

Zero-to-peak height of a waveform in nanometers.

analyst

Personnel responsible for reviewing and revising the results of automatic processing.

ANSI

American National Standards Institute.

array

Collection of sensors distributed over a finite area (usually in a cross, triangle, or concentric pattern) and referred to as a single station.

arrival

Detected signal that has been associated to an event. First, the Global Association (GA) software associates the detection to an event. Later, during interactive processing, many arrivals are confirmed, improved, or added by visual inspection.

arrival-quality test

GA test of an event's quality based on the value of the slowness uncertainty and the distance between the event and station for each defining arrival.

arrival tag

Record in the **ga_tag** table that characterizes an arrival to support logic in *GAassoc* and *GAconflict*, for instance as `REQUESTED` for auxiliary seismic arrivals.

associate

Assign an arrival to an S/H/I event.

associated phase

Phase that is associated with an S/H/I event.

association set

Set of arrivals associated with an event hypothesis or confirmed event in GA.

attribute

(1) Database column. (2) Characteristic of an item; specifically, a quantitative measure of a S/H/I detection such as azimuth, slowness, period, or amplitude.

azimuth

Direction, in degrees clockwise with respect to North, from a station to an event.

B

b value

Slope of the line fit to a plot of seismic magnitude versus cumulative number of events, usually computed for a finite geographic area.

beam

(1) Waveform created from array station elements that are sequentially summed after being steered to the direction of a specified azimuth and slowness. (2) Any derived waveform (for example, a filtered waveform).

build

(1) To create an event by detecting its seismic or hydroacoustic signals, associating its arrivals, identifying them as phases, and locating the event. (2) An operational version of a system or component that incorporates a specified subset of the capabilities that the final product will provide.

bulletin

Chronological listing of event origins spanning an interval of time. Often, the specification of each origin or event is accompanied by the event's arrivals and sometimes with the event's waveforms.

C

CD-ROM

Compact Disk–Read Only Memory.

channel

Component of motion or distinct stream of data.

CMR

Center for Monitoring Research.

command

Expression that can be input to a computer system to initiate an action or affect the execution of a computer program.

component

(1) One dimension of a three-dimensional signal; (2) The vertically or horizontally oriented (north or east) sensor of a station used to measure the dimension; (3) One of the parts of a system; also referred to as a module or unit.

Comprehensive Nuclear-Test-Ban Treaty Organization

Treaty User group that consists of the Conference of States Parties (CSP), the Executive Council, and the Technical Secretariat.

Computer Software Component

Functionally or logically distinct part of a computer software configuration item; possibly an aggregate of two or more software units.

Computer Software Configuration Item

Aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process.

confirmed event

An event that passed the minimum weighted count threshold test, an arrival quality test, and a probability of detection test in GA.

conflict resolution

GA process by which arrivals, initially associated to multiple events, are disassociated from all but one of the events.

corroborating arrival

Arrival that is added to an event seeded by the driver arrival that helps to corroborate the preliminary event hypothesis in GA.

CSC

See Computer Software Component.

CSCI

See Computer Software Configuration Item.

CTBTO

See Comprehensive Nuclear-Test-Ban Treaty Organization.

D

DACS

Distributed Application Control System. This software supports inter-application message passing and process management.

defining

Arrival attribute, such as arrival time, azimuth, or slowness, which is used in calculating the event's location or magnitude.

defining arrival

Arrival whose attributes (time, azimuth, and/or slowness) are used to compute an event location.

defining phase

Associated phase for which features are used in the estimation of the location and origin time of an S/H/I event.

deg.

Degrees (as a distance).

detection

Probable signal that has been automatically detected by the Detection and Feature Extraction (*DFX*) software.

DFX

Detection and Feature Extraction. *DFX* is a programming environment that executes applications written in Scheme (known as *DFX* applications).

driver arrival

Arrival that is used as an initial seed to build an automatic event in GA. This arrival was detected at one of the stations close to the grid cell being evaluated.

E

event

Unique source of seismic, hydroacoustic, or infrasonic wave energy that is limited in both time and space.

event hypothesis

Association set that is not yet located or confirmed by the Location and Confirmation process in GA.

execute

Carry out an instruction, process, or computer program.

F

f-k

Frequency versus wavenumber (k) analysis that maps phase power from an array as a function of azimuth and slowness.

FTP

File Transfer Protocol; protocol for transferring files between computers.

G

GA

Global Association application. GA associates S/H/I phases to events.

GAcons

GA application that precomputes propagation knowledge base information and stores it in two grid files used by GA.

GB

Gigabyte. A measure of computer memory or disk space that is equal to 1,024 megabytes.

grid

Set of points used by GA covering either a region of the earth or the whole earth and including the interior where deep seismicity occurs. Information about propagation to a network of stations is computed by *GAcons* for a grid and stored in a binary file.

grid cell

Volume within the earth around a grid point in the GA grid that is characterized by the grid point location, a radius, and a depth range around that grid point.

grid point

Location (latitude, longitude, and depth) on the grid used by GA to perform its exhaustive association set search.

GSETT-3

Group of Scientific Experts Third Technical Test.

GUI

Graphical User Interface.

H

hydroacoustic

Pertaining to sound in the ocean.

I

IDC

International Data Centre.

IMS

International Monitoring System.

infrasonic (infrasound)

Pertaining to low-frequency (sub-audible) sound in the atmosphere.

instance

Running computer program. An individual program may have multiple instances on one or more host computers.

IPC

Interprocess communication. The messaging system by which applications communicate with each other through *libipc* common library functions. See *tuxshell*.

J

jdate

Modified Julian Date. Concatenation of the year and three-digit Julian day of year. For example, the jdate for 07 March , 2000 is 2000067.

Julian date

Increasing count of the number of days since an arbitrary starting date.

K

km

Kilometer.

L

LAN

Local Area Network.

lat.

Latitude.

M

magnitude

Empirical measure of the size of an event (usually made on a log scale).

MB

Megabyte. 1,024 kilobytes.

m_b

Magnitude estimated from seismic body waves.

M_L

Magnitude estimated from seismic waves measured near the source.

N

network

Spatially distributed collection of seismic, hydroacoustic, or infrasonic stations for which the station spacing is much larger than a wavelength.

network processing

Processing that uses the results of Station Processing from a network of stations to define and locate events.

noise

Incoherent natural or artificial perturbations of the waveform trace caused by ice, animals migrations, cultural activity, equipment malfunctions or interruption of satellite communication, or ambient background movements.

O

ORACLE

Vendor of the database management system used at the PIDC and IDC.

orid

Origin Identifier.

origin

Hypothesized time and location of a seismic, hydroacoustic, or infrasonic event. Any event may have many origins. Characteristics such as magnitudes and error estimates may be associated with an origin.

P

parameter

User-specified token that controls some aspect of an application (for example, database name, threshold value). Most parameters are specified using [*token* = *value*] strings, for example, `dbname=mydata/base@oracle`.

parameter (par) file

ASCII file containing values for parameters of a program. Par files are used to replace command line arguments. The files are formatted as a list of [*token* = *value*] strings.

phase

Arrival that is identified based on its path through the earth.

phase name

Name assigned to a seismic, hydroacoustic or infrasonic arrival associated with a travel path.

PIDC

Prototype International Data Centre.

pipeline

1) Flow of data at the IDC from the receipt of communications to the final automated processed data before analyst review. 2) Sequence of IDC processes controlled by the DACS that

either produce a specific product (such as a Standard Event List) or perform a general task (such as station processing).

PKB grid file

Propagation Knowledge-base grid file. This (GA) grid file contains the travel-time and propagation characteristics for all stations in the network and all grid cells (see grid and grid cell).

post-analysis processing

Automated processing that occurs after analysts have reviewed the automatic event bulletins.

post-location processing

Software that computes various magnitude estimates and selects data to be retrieved from auxiliary stations.

primary phase

First arriving phase recorded at a S/H/I station.

primary seismic

IMS seismic station(s) or data that is (are) part of the detection network.

primary stations

Stations that make up the primary seismic network of the IMS. Primary stations send data continuously to the IDC.

process

Function or set of functions in an application that perform a task.

program

Organized list of instructions that, when executed, causes the computer to behave in a predetermined manner. A program contains a list of variables and a list of statements that tell the computer what to do with the variables.

R

RAM

Random Access Memory.

RDBMS

Relational Database Management System.

regional

(1) (distance) Source-to-seismometer separations between a few degrees and 20 degrees. (2) (event) Recorded at distances where the first P and S waves from shallow events have traveled along paths through the uppermost mantle.

S

s

Second(s) (time).

s.d.

Standard deviations.

S/H/I

Seismic, hydroacoustic, and infrasonic.

SAIC

Science Applications International Corporation.

SASC

Slowness-Azimuth Station Corrections.

script

Small executable program, written with UNIX and other related commands, that does not need to be compiled.

secondary phases

Phases that arrive after the primary phase.

SEL1

Standard Event List 1; S/H/I bulletin created by total automatic analysis of continuous timeseries data. Typically, the list runs one hour behind real time.

SEL2

Standard Event List 2; S/H/I bulletin created by totally automatic analysis of both continuous data and segments of data specifically down-loaded from stations of the auxiliary seismic network. Typically, the list runs five hours behind real time.

SEL3

Standard Event List 3; S/H/I bulletin created by totally automatic analysis of both continuous data and segments of data specifically down-loaded from stations of the auxiliary seismic network. Typically, the list runs 12 hours behind real time.

seismic

Pertaining to elastic waves traveling through the earth.

S/H/I

Seismic, hydroacoustic, and infrasonic.

slowness

Inverse of velocity, in seconds/degree; a large slowness has a low velocity.

slowness vector

Vector in 2-D wavenumber space. The magnitude of the vector corresponds to the inverse of the phase velocity of a traveling plane wave. The direction of the vector is usually defined as being from the station to the source.

snr

Signal-to-noise ratio.

split event

Event that has been incorrectly formed by GA as several events that associate subsets of the arrivals from the actual event.

SQL

Structured Query Language; a language for manipulating data in a relational database.

station

Collection of one or more monitoring instruments. Stations can have either one sensor location (for example, BGCA) or a spatially distributed array of sensors (for example, ASAR).

station processing

Processing based on data from a single station.

T

tar

Tape archive. UNIX command for storing or retrieving files and directories. Also used to describe the file or tape that contains the archived information.

teleseismic

1) (distance) Source to seismometer separations of 20 degrees or more. (2) (event) Recorded at distances where the first P and S waves from shallow events have traveled paths through the mantle/core.

Tuxedo

Transactions for UNIX Extended for Distributed Operations.

tuxshell

Process in the Distributed Processing CSCI used to execute and manage applications. See IPC.

U

UNIX

Trade name of the operating system used by the Sun workstations.

W

waveform

Time-domain signal data from a sensor (the voltage output) where the voltage has been converted to a digital count (which is monotonic with the amplitude of the stimulus to which the sensor responds).

WorkFlow

Software that displays the progress of automated processing systems.

Index

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

Z